Implementation of Tic-Tac-Toe Game Using Python and Tkinter on Raspberry Pi

Gayatri Tangudu
Student ID - 40221830
Information Systems Engineering
Concordia University
Montreal, Canada
gayatri.tangudu@concordia.ca

December 2023

Abstract

In order to create an interactive Tic-Tac-Toe game, this project aims to combine the capabilities of the Raspberry Pi, Python, and Tkinter. Connecting the dots between physical computing and programming instruction, the Raspberry Pi's smooth Python integration offers hobbyists an adaptable platform. Selecting a game that combines strategic complexity with ease of use, Tic-Tac-Toe pits two players against one another to strategically mark a three-by-three grid with the letters 'X' and 'O.' The project painstakingly puts together the necessary parts, using classes such as TicTacToeBoard and TicTacToeGame for the structural logic and visual design of the game. The game board appears as a dynamic interface using Tkinter widgets, utilizing a model-view-controller architecture. A Tkinter is used to transmit game status updates in real time.Label when using a tkinter-facilitated interactive cell grid.Button widgets that record and decipher player movements. Using Python and Tkinter, an immersive investigation of programming, gaming, and physical computing is encapsulated in this abstract, which is a Tic-Tac-Toe game created on a Raspberry Pi.

Contents

List of Figures

Chapter 1

Introduction

1.1 Project Overview

Among physical computer boards, the Raspberry Pi is a platform that is popular with both amateurs and novices because it is so flexible. It allows for easy interaction with Python and may be used for a variety of applications, from instructional projects to do-it-yourself projects. The goal of this project is to use Python and the Tkinter GUI toolkit to build a Tic-Tac-Toe game by utilizing the Raspberry Pi's capabilities [4].

The simplicity of Tic-Tac-Toe makes it a great starting point for programming aficionados who want to get into game creation. The game basically consists of two players strategically placing their marks on a three-by-three grid in a strategic fight. Each player is given the role of ”X” or ”O.” When three markers are effectively aligned horizontally, vertically, or diagonally, the player wins. When every cell is marked, the game ends graciously in a tie if there isn't a clear winner.

Starting this project entails coordinating important game elements. The TicTacToeBoard class is responsible for creating the game board, while the TicTacToeGame class is in charge of handling the game logic. By utilizing a Tkinter window, the game board transforms into a dynamic interface by utilizing a model-view-controller architecture. This window features two essential components:

Top Display: Providing instantaneous glimpses into the story as it develops in the game.

Grid of Cells: Marking accessible and marked places, providing a visual depiction of players' movements.

Visualization is achieved by skillfully using Tkinter widgets[5]. A tkinter is used to curate the top display.Label, as a tkinter array causes the cell grid to take shape.widgets with buttons. The latter becomes an interactive canvas that, when a player engages it to assess movements and determine the game's conclusion, activates the logic of the game. As the model, the game logic skillfully handles the complexities of the data, rules, and overall game flow.

1.2 Motivation

This project was chosen because it has two important applications: it will enhance teaching, and it will allow me to create a fun game just for the Raspberry Pi platform. The opportunity to dive into the nuances of Python programming and GUI creation using Tkinter is highlighted by the educational value, providing a hands-on learning experience. Investigating the incorporation of hardware—more especially, the Raspberry Pi—enhances the instructional value by offering insights into embedded systems and physical computing.

The attraction of developing an engaging and interactive game simultaneously gives the project a thrilling new dimension. Not only does designing a Tic-Tac-Toe game demonstrate programming skill, but it also gives one the joy of turning abstract concepts into a real, playable product. since of its adaptability, the Raspberry Pi is a perfect platform for this kind of work since it can combine hardware and software components in a way that is both small and easy to use.

Basically, the motivation is the desire to learn and employ programming abilities in a real-world setting, together with the excitement of creating a game that is entertaining and instructive for users of all skill levels and adding to the thriving ecosystem of the Raspberry Pi.

1.3 Objectives

The principal aims of this project are to create a playable and intuitive Tic-Tac-Toe game on the Raspberry Pi platform. These goals are carefully constructed to guarantee a thorough and satisfying project conclusion. The main objectives are as follows:

Functional Implementation: Using the Raspberry Pi's Python and Tkinter capabilities, we want to develop a stable and complete Tic-Tac-Toe game. This entails putting in place a sensible user interface, precise game mechanics, and a coherent game structure.

User-Friendly Design: To put the user's experience first by creating a visually appealing and simple gaming interface. This includes the smooth operation of the game, the unambiguous display of the current state of the game, and the user-friendly interactions that improve gaming in general.

Raspberry Pi Integration: Utilizing the Raspberry Pi platform's hardware capabilities, to smoothly integrate the game with it. In order to do this, the game must be optimized for the Raspberry Pi platform and made to take use of Python's compatibility with the environment.

Accessibility: To ensure that the Tic-Tac-Toe game is accessible to users of varying skill levels. This involves implementing clear instructions, user prompts, and error handling to enhance the overall accessibility and inclusivity of the gaming experience.

By accomplishing these objectives, the project aims to deliver not only a successful implementation of a classic game but also a valuable learning resource for those exploring the realms of programming and game development on the Raspberry Pi platform.

Chapter 2

Background

2.1 Overview of Tic-Tac-Toe

Take a fascinating trip as we explore how to create a Tic-Tac-Toe game in Python. This project demonstrates how to create an interactive graphical user interface (GUI) for our game by using the Tkinter toolkit from the Python standard library. The sample image that goes with this project shows off the intended outcome and gives an idea of how the finished output will look[3].

With an intuitive interface, our Tic-Tac-Toe game perfectly replicates the classic three-by-three gaming board. In turns, players will place their marks strategically on a shared gadget. Located prominently at the top of the window, the game display changes dynamically to show the player in charge at any given time.

The game display will joyfully declare the winner by displaying their mark (X or O) or name when they have won. In addition, to provide even more excitement, the winning combination of cells will be visibly highlighted on the game board at the same time.

The game features a File menu that allows players to reset the game for a rematch or to end their gaming session graciously, which improves user experience.

Let's go on this coding trip together if you're excited by the idea of utilizing Python and Tkinter to create a dynamic and eye-catching Tic-Tac-Toe game!

Figure 2.1: Overview of Tic-Tac-Toe.

2.2 Introduction to Python, Tkinter, and Raspberry Pi

2.2.1 Python:

Python is a well-known, high-level programming language that is easy to read and understand. Because of its syntax, which prioritizes code readability, it is a popular choice for a wide range of domains and is perfect for novices. Python's popularity may be attributed to its large standard library and active community, which allow developers to work effectively on a wide range of jobs. Python continues to be a strong and useful tool for a wide range of applications, from web development to data science and beyond[1].

2.2.2 Tkinter:

Python's default GUI (Graphical User Interface) toolkit, Tkinter, is a natural extension of the language. It's a great option for both new and seasoned developers since it provides an easy-to-use and simple way to create graphical interfaces. Tkinter makes it easier to create interactive programs by offering a variety of widgets for creating user interfaces. Tkinter's cross-platform compatibility guarantees consistent operation of programs across a range of operating systems [5].

2.2.3 Raspberry Pi:

The Raspberry Pi, a small single-board computer, has become a popular platform for professionals, students, and amateurs alike. Its little size, comparable to that of a credit card, belies a wide range of features, such as GPIO (General Purpose Input/Output) pins, which makes it the perfect option for experimentation and hardware projects. The Raspberry Pi, which runs on many operating systems, enables a wide range of applications, including robotics and home automation. Its versatility, affordability, and accessibility have made it a game-changer for physical computing and embedded systems education and implementation. The combination of Python, Tkinter, and Raspberry Pi creates a powerful ecosystem for creating interesting projects that combine hardware and software elements in a seamless manner [4].

Chapter 3

System Architecture

3.1 Hardware Components

3.1.1 Required Hardware

The following hardware is required for the initial setup of your Raspberry Pi :

Monitor We will need a monitor during the initial setup and configuration of the operating system.

microSD Card Raspberry Pi uses a microSD card to store the operating system and files.

Keyboard and Mouse A USB keyboard and mouse are required during the initial setup of the Raspberry Pi. Once the setup is complete, we can switch to using Bluetooth versions of these peripherals.

HDMI Cables and Power Supply: We need an HDMI cable to connect the Raspberry Pi to a monitor.The Raspberry Pi uses a USB connection to power the board. Again, different Raspberry Pi models have different USB connection and power requirements.

3.1.2 Raspberry Pi Board Overview:

The following parts are present on the Raspberry Pi 3 B+ board:

General-purpose input-output pins: The Raspberry Pi is connected to electronic parts via these pins.

Ethernet port: The Raspberry Pi is connected to a wired network using this connector. In addition, the Raspberry Pi features integrated Bluetooth and Wi-Fi for wireless connectivity.

Two USB 3.0 and two USB 2.0 ports: These USB ports are used to connect peripherals like a keyboard or mouse. The two black ports are USB 2.0 and the two blue ports are USB 3.0.

AV jack: This AV jack allows you to connect speakers or headphones to the Raspberry Pi.

Camera Module port: This port is used to connect the official Raspberry Pi Camera Module, which enables the Raspberry Pi to capture images.

HDMI ports: These HDMI ports connect the Raspberry Pi to external monitors. The Raspberry Pi 4 features two micro HDMI ports, allowing it to drive two separate monitors at the same time.

USB power port: This USB port powers the Raspberry Pi. The Raspberry Pi 4 has a USB Type-C port, while older versions of the Pi have a micro-USB port.

External display port: This port is used to connect the official seven-inch Raspberry Pi touch display for touch-based input on the Raspberry Pi.

microSD card slot (underside of the board): This card slot is for the microSD card that contains the Raspberry Pi operating system and files.

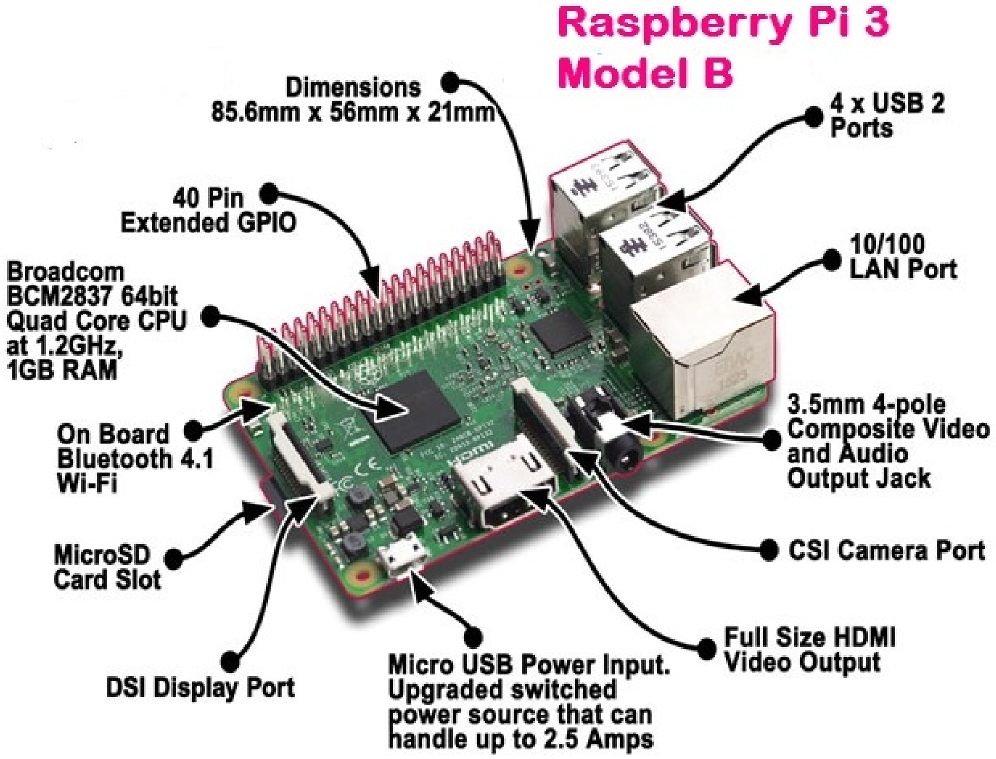


Figure 3.1: Raspberry Pi Board Overview.

Chapter 4

Implementation

4.1 Setting Up the Raspberry Pi

This step is to connect everything and set up the operating system now that the microSD card and other components are ready. First, let's connect each and every peripheral:

1. Place the microSD card into the Raspberry Pi's bottom card slot.
2. Use any one of the four USB ports to connect the mouse and keyboard.
3. Using an HDMI cable designed specifically for your Raspberry Pi model. Connect a monitor to one of the HDMI ports.
4. Link a power source to the USB power outlet.

After connecting the peripherals, turn on your Raspberry Pi to set up the operating system [4].

4.1.1 Setup Wizard

Upon initial boot, Raspbian offers an installation process that assists with setting up your Wi-Fi network, password, locale, and operating system update. Proceed and carry out these actions as directed[4].

We may start learning Python on the Raspberry Pi by restarting the operating system when we have finished the instructions!

4.2 Running Python on the Raspberry Pi

Python is a first-class citizen on the Raspberry Pi, which is one of the greatest things about working with it. Python was chosen as the primary language by the Raspberry Pi Foundation due to its strength, adaptability, and user-friendliness. Raspbian ships with Python preloaded, so we'll be good to go right away [4].

When it comes to programming Python on the Raspberry Pi, we have a wide range of choices. Throughout the project, we will utilize the widely-liked option is the Mu editor.

4.3 Step by Step Guide for the Game Logic

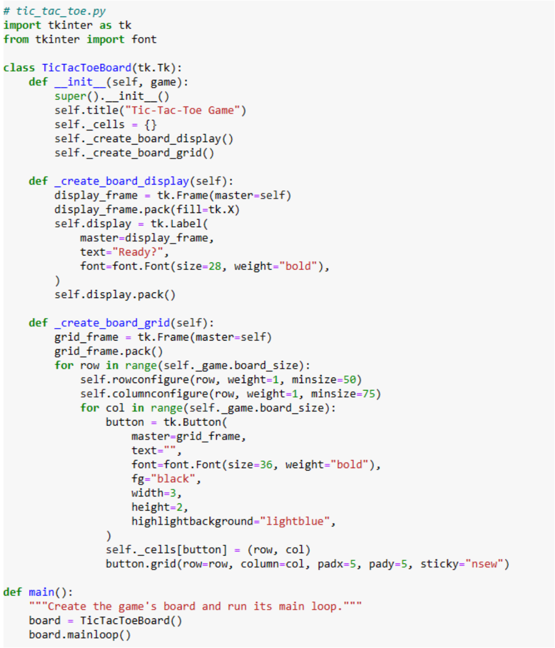
4.3.1 Step 1: Set Up the Tic-Tac-Toe Game Board With Tkinter

We will utilize a regular Python installation to do this project. Since there is no requirement for an external reliance, there is no need to construct a virtual environment. Tkinter is the only package we will require, and it is included in the Python standard library[3].

Create a Class to Represent the Game Board:

Using the Tk class, which lets you make the main window of our Tkinter application, we may construct the board for our game of tic tac toe. Next, we will add a grid of cells to fill the remaining area of the main window and a display to the top frame.

Utilizing Button objects, created the three-by-three cell grid to expand the game display. This technique forms the basis of the Tic-Tac-Toe game board inside the graphical user interface by dynamically generating the interactive canvas on which players can make their movements.



```python
# tic_tac_toe.py
import tkinter as tk
from tkinter import font

class TicTacToeBoard(tk.Tk):
    def __init__(self, game):
        super().__init__()
        self.title("Tic-Tac-Toe Game")
        self._cells = {}
        self._create_board_display()
        self._create_board_grid()

    def _create_board_display(self):
        display_frame = tk.Frame(master=self)
        display_frame.pack(fill=tk.X)
        self.display = tk.Label(
            master=display_frame,
            text="Ready?",
            font=font.Font(size=28, weight="bold"),
        )
        self.display.pack()

    def _create_board_grid(self):
        grid_frame = tk.Frame(master=self)
        grid_frame.pack()
        for row in range(self._game.board_size):
            self.rowconfigure(row, weight=1, minsize=50)
            self.columnconfigure(row, weight=1, minsize=75)
            for col in range(self._game.board_size):
                button = tk.Button(
                    master=grid_frame,
                    text="",
                    font=font.Font(size=36, weight="bold"),
                    fg="black",
                    width=3,
                    height=2,
                    highlightbackground="lightblue",
                )
                self._cells[button] = (row, col)
                button.grid(row=row, column=col, padx=5, pady=5, sticky="nsew")

def main():
    """Create the game's board and run its main loop."""
    board = TicTacToeBoard()
    board.mainloop()
```

Figure 4.1: Tic-Tac-Toe Game Board With Tkinter Code

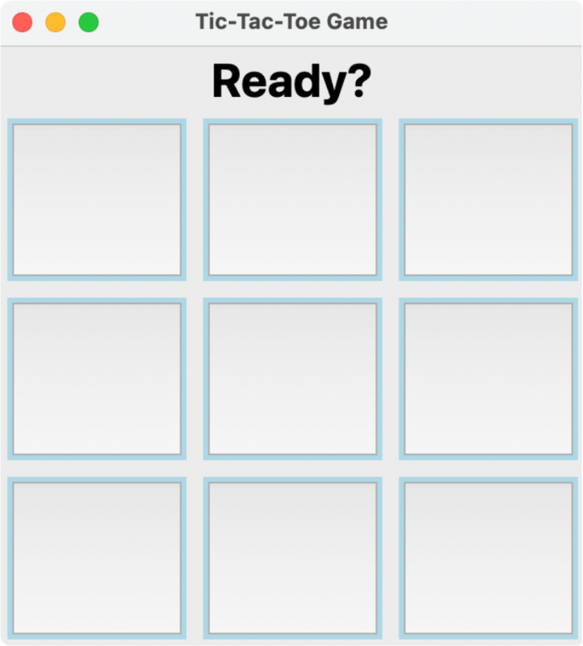


Figure 4.2: Tic-Tac-Toe Game Board With Tkinter

4.3.2 Step 2: Set Up the Tic-Tac-Toe Game Logic in Python

This section describes the fundamental principles and procedures that underpin the game of Tic Tac Toe. The first comprehension builds a sublist of coordinates by iterating across the grid's rows, obtaining the coordinates of each cell. A winning combination is represented by each sublist of coordinates. The coordinates of every cell in the grid columns are created as sublists by the second comprehension.

A similar method is used in the third and fourth comprehensions to obtain the coordinates of each cell in the board diagonals. Lastly, a list of lists with every conceivable winning combination on the tic tac toe board is returned by the technique.

You may now consider processing the movements made by the players after setting up your game board.

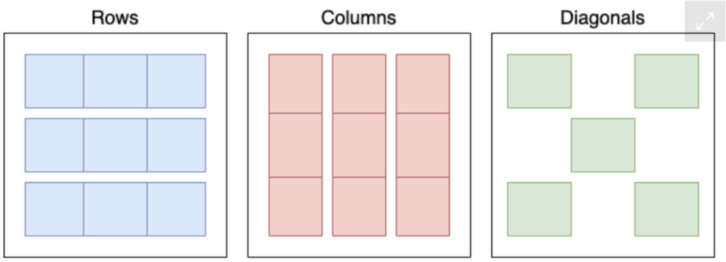


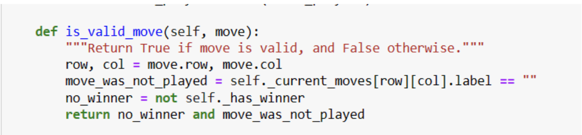
Figure 4.3: Figure Out the Winning Combinations.

4.3.3 Step 3: Process the Players' Moves on the Game's Logic

We will primarily deal with one kind of occurrence in this game of tic tac toe: player movements. To put it in Tkinter language, a player can move by simply clicking on the button widget that represents the selected cell [5].

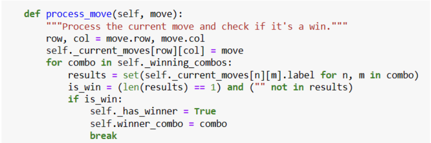
The TicTacToeGame class will go through several operations in response to each player's move. Among these surgeries are:

- Validating the move
- Checking for a winner
- Checking for a tied game
- Toggling the player for the next move



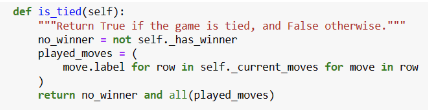
```
def is_valid_move(self, move):
    """Return True if move is valid, and False otherwise."""
    row, col = move.row, move.col
    move_was_not_played = self._current_moves[row][col].label == ""
    no_winner = not self._has_winner
    return no_winner and move_was_not_played
```

Figure 4.4: Validating the move



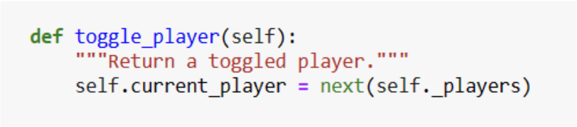
```
def process_move(self, move):
    """Process the current move and check if it's a win."""
    row, col = move.row, move.col
    self._current_moves[row][col] = move
    for combo in self._winning_combos:
        results = set(self._current_moves[n][m].label for n, m in combo)
        is_win = (len(results) == 1) and ("" not in results)
        if is_win:
            self._has_winner = True
            self.winner_combo = combo
            break
```

Figure 4.5: Checking for a winner



```python
def is_tied(self):
    """Return True if the game is tied, and False otherwise."""
    no_winner = not self._has_winner
    played_moves = (
        move.label for row in self._current_moves for move in row
    )
    return no_winner and all(played_moves)
```

Figure 4.6: Checking for a tied game



```python
def toggle_player(self):
    """Return a toggled player."""
    self.current_player = next(self._players)
```

Figure 4.7: Toggling the player for the next move

4.3.4 Step 4: Process Players’ Moves on the Game Board

We can now manage the players’ actions based on the logic of the game. It is now necessary to link this reasoning to the actual game board. In order for the board to react to player movements, we must also develop the programming. We can play the game for first time now.

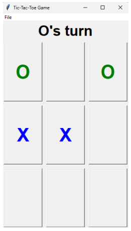


Figure 4.8: Process Players’ Moves on the Game Board.

4.3.5 Step 5: Provide Options to Play Again and Exit the Game

We’ll give the primary menu for our tic tac toe game in this part. The option to resume the game will be available on this screen, allowing players to begin a new match. After the players have completed the game, it will also offer the option to quit [2].

In many GUI programs, main menus are a crucial component. Therefore, developing them using Tkinter is a useful exercise to enhance our GUI-related abilities outside of game production.

Here’s an illustration of how creating your own games can be a highly educational experience as it lets you incorporate abilities and information into projects that aren’t games.

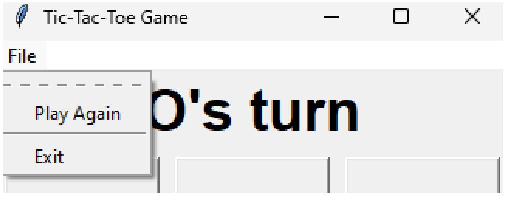


Figure 4.9: Options to Play Again and Exit the Game

Chapter 5

Conclusion

In this practical project, we have used the dynamic combination of Python, Tkinter, and Raspberry Pi gives us the ability to create games and opens us a world of creative possibilities. This project acts as a springboard, launching us into an innovative realm where hardware and code collide and laying the groundwork for fascinating future projects.

Along the way, we've refined a few crucial abilities:

- **Implementation of Game Logic:** Proficiency in implementing the complex logic underlying the beloved game tic tac toe, therefore strengthening our foundation in Python game programming.

- **Tkinter GUI Construction:** Skill in using Tkinter to create a visually appealing and user-friendly gaming board that improves the user experience overall with a well-designed graphical user interface.

- **Logic-GUI Integration:** The ability to seamlessly combine the logic of a game with its graphical user interface is essential for developing interactive applications that have user actions that make sense in relation to the underlying program logic.

- **Exploration with Raspberry Pi:** By delving into the world of Raspberry Pi, we have discovered how to run and modify our code for this flexible operating system. This experience extends our knowledge base and shows how flexible our abilities are in a range of hardware configurations.

- **Proficiency with Mu Editor:** Gaining knowledge of the Mu editor for Raspberry Pi, an essential tool for developing and exploring this distinctive environment.

Chapter 6

Future Work

6.1 Future Work:Elevating Tic-Tac-Toe Game Project

Although the existing Tic-Tac-Toe implementation is a good starting point, there are several intriguing potential improvements that may be made in the future:

- **Customizable Board Sizes:** Permit users to choose the size of the game board and add different degrees of difficulty to their play experience.
- **Computer Opponent with Difficulty Levels:** Introduce an AI opponent that may be adjusted in difficulty to provide gamers with a demanding solo experience.
- **Online Multiplayer Mode:** Include an online multiplayer component to broaden the game's appeal and encourage participation and competitiveness across borders.
- **Alternate Markers and Themes:** Add variation to the game by offering users the ability to customize it to their likings with changeable markers and theme options.
- **Persistent User Profiles:** Establish user profiles with customized data and settings to promote long-term interaction and a feeling of continuity.

With these improvements, the game Tic-Tac-Toe will become a dynamic, feature-rich program that provides players with a more engaging and customized gaming experience.

Bibliography

[1] Learning python — the hitchhiker's guide to python. https://docs.python-guide.org/intro/learning/, 2023. Accessed: 2023-12-03.

[2] Tic tac toe game with gui using tkinter in python - geeksforgeeks. https://www.geeksforgeeks.org/tic-tac-toe-game-with-gui-using-tkinter-in-python/, 2023. Accessed: 2023-12-03.

[3] Real Python. Build a tic-tac-toe game with python and tkinter – real python. https://realpython.com/tic-tac-toe-python/#project-overview, 2023. Accessed: 2023-12-03.

[4] Real Python. Build physical projects with python on the raspberry pi – real python. https://realpython.com/python-raspberry-pi/, 2023. Accessed: 2023-12-03.

[5] Real Python. Python gui programming with tkinter – real python. https://realpython.com/python-gui-tkinter/, 2023. Accessed: 2023-12-03.