REPORT OF THE PROJECT

Smart Contact Manager

Submitted by

Ankita Kumari(24MCA20142)

Submitteted to

Shivam Sharma

In partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATION

IN

University Institute of Computing (UIC)

CHANDIGARH UNIVERSITY

SESSION 2024 -26

Table of Contents

---

1. Introduction

The **Tic Tac Toe Game** is a two-player game that involves marking a 3x3 grid with symbols (typically "X" and "O") in turns, aiming to complete a line of three symbols either horizontally, vertically, or diagonally. This project implements the game in Python as a console application, using text-based interaction.

The game is designed to allow players to alternate turns, check for win conditions, and detect a draw when all cells are occupied without any winner. The primary goal of this project is to practice basic programming constructs and apply them in the development of a simple game.

2. Project Objectives

- **Develop a fully functional Tic Tac Toe game** that is intuitive and interactive.
- **Implement fundamental programming concepts** such as loops, conditionals, functions, and lists.
- **Ensure accurate input validation** and logical flow for seamless gameplay.
- **Enhance understanding of modular programming** through the use of well-defined functions and a structured approach.
- **Allow for replayability**, giving users an option to restart the game after each session.

3. Software Development Life Cycle (SDLC)

The SDLC followed for this project is based on the **Waterfall Model**, a sequential approach ideal for small-scale projects with well-defined requirements.

Phases:

1. **Requirement Analysis**: Defining the game rules and technical requirements.
2. **System Design**: Planning the data structures, algorithms, and function modules for the game.
3. **Implementation**: Coding the game in Python using the designed structure.
4. **Testing**: Validating each function, as well as the overall game, for correctness and robustness.
5. **Deployment**: Running the game in a Python environment for user testing.
6. **Maintenance**: Documenting the project and identifying potential future enhancements.

4. Project Planning and Scope

4.1 Project Timeline

- **Week 1**: Requirement gathering and design.
- **Week 2**: Initial coding and development of core functions.
- **Week 3**: Testing, debugging, and validation.
- **Week 4**: Documentation and finalization.

4.2 Scope of the Project

- **In Scope**: Basic two-player mode, text-based interaction, core game logic (winning and draw conditions).
- **Out of Scope**: Advanced GUI, AI player, multiplayer over a network.

5. Requirements Analysis

5.1 Functional Requirements

1. **Player Interaction**: Enable two players to take turns selecting positions on the grid.
2. **Game Board Display**: Show a clear representation of the board after every move.
3. **Win Conditions**: Check if a player has won after each move.
4. **Draw Condition**: Detect and declare a draw if all spaces are filled without a winner.
5. **Replay Option**: Allow players to start a new game without restarting the application.

5.2 Non-Functional Requirements

1. **User-Friendliness**: Ensure easy-to-read instructions and board display.
2. **Efficiency**: Optimize code for smooth and quick gameplay.
3. **Maintainability**: Use modular functions to make the code easy to understand and modify.

5.3 Technical Requirements

- **Programming Language**: Python 3.x
- **IDE/Environment**: Any Python-compatible environment such as VS Code, PyCharm, or Jupyter Notebook.

6. System Design and Architecture

The Tic Tac Toe game design is based on a modular, functional approach to ensure simplicity and maintainability. The main components are:

6.1 Game Board Representation

- The board is represented as a 3x3 grid using a list with nine elements, each representing a cell in the grid.

6.2 Core Functions

1. **Display Board**: Renders the current state of the board for players.
2. **Player Input**: Takes and validates user input to ensure the selected position is within the grid and not occupied.
3. **Win Check**: Checks rows, columns, and diagonals for a winning pattern.
4. **Draw Check**: Verifies if the board is full and no winning condition is met.
5. **Turn Switch**: Alternates turns between Player X and Player O.

6.3 Data Flow Diagram

plaintext
Copy code
User Input -> Validate Input -> Update Board -> Check Win/Draw -> Display Board -> Switch Player

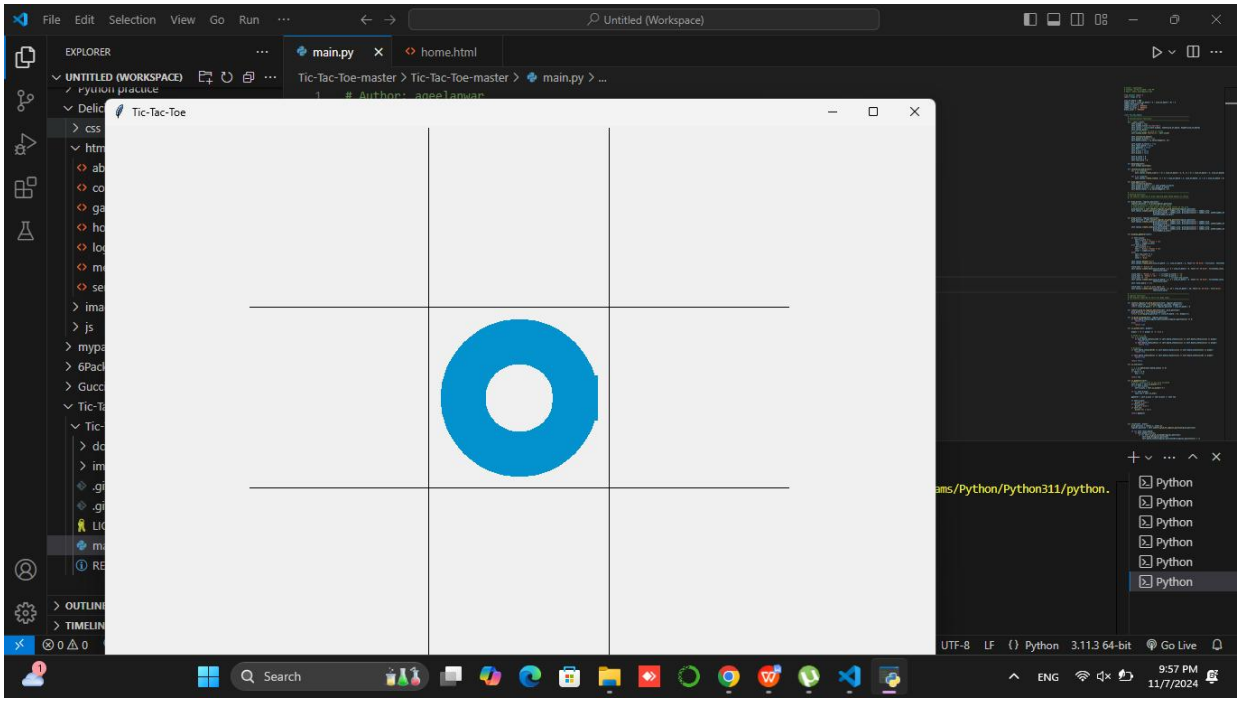
6.4 Pseudo Code

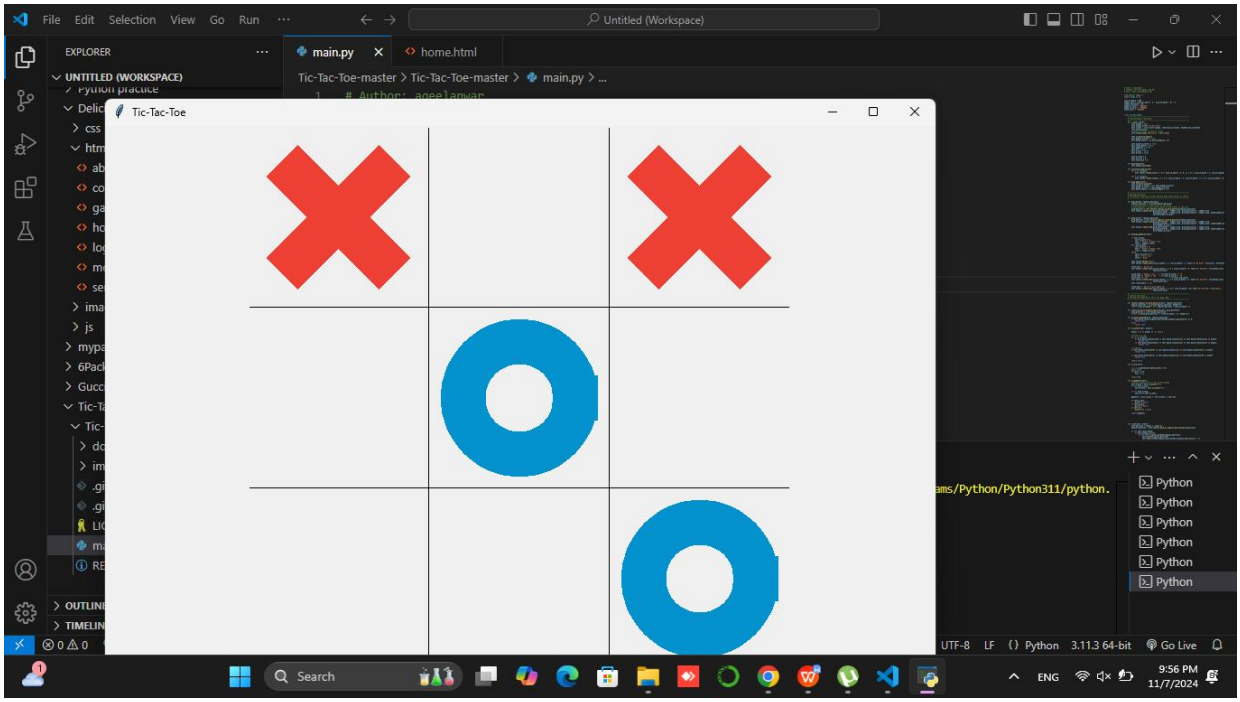
- Initialize an empty board as a 3x3 grid.
- While the game is active:
    - Display the board.
    - Prompt the current player for input.
    - Validate the input and update the board.
    - Check if the current move resulted in a win or if the board is full (draw).
    - Switch players if no win or draw is detected.

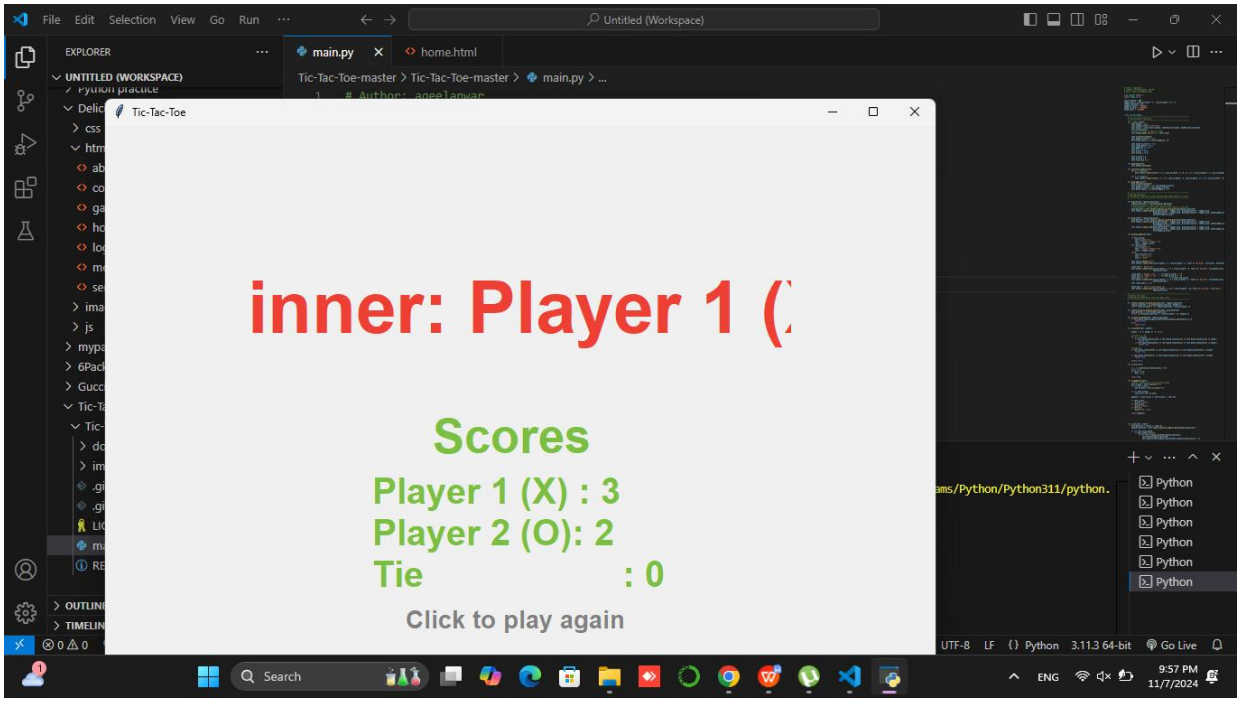
7. Results

The Tic Tac Toe game performs accurately, displaying the board after each move and correctly identifying win and draw scenarios. All test cases were successfully passed, ensuring robust gameplay.

Some screenshorts of the project are:-







8. Challenges and Solutions

Challenge 1: Input Validation

- **Solution**: Implemented a function to validate that input is within range and not occupied.

Challenge 2: Modular Code Organization

- **Solution**: Used multiple functions to split game logic, which improved readability and maintainability.

Challenge 3: Winning Logic

- **Solution**: Developed a function to compare board indices and ensure all winning conditions are checked in each turn.

9. Future Enhancements

- **Graphical Interface**: Add a GUI using Tkinter or Pygame for a more visually appealing version.
- **AI Opponent**: Implement an AI-based player that uses the Minimax algorithm to make strategic moves.
- **Score Tracking**: Record scores over multiple games to keep track of wins, losses, and draws.

10. Conclusion

The **Tic Tac Toe Game** project effectively showcases fundamental Python programming skills, providing a straightforward yet engaging experience. By implementing a text-based, turn-based game with clear rules, winning conditions, and draw detection, this project helps to solidify foundational concepts such as loops, functions, and conditional logic. The modular design allows easy adjustments or extensions, making it a valuable learning tool for beginners in programming.

This project serves as a practical introduction to game development, offering insight into logical thinking, structured programming, and user interaction. It's a small but illustrative example of how a simple idea can evolve with programming principles, creating a functional, interactive application. With potential future enhancements such as a graphical interface, AI opponent, and score tracking, the game could become even more engaging and educational, offering a comprehensive foundation in Python for beginners and intermediate learners alike.

11. References

1. **Python Programming Language**
   Python Software Foundation. Official website and documentation for Python: https://www.python.org.
2. **Tkinter Documentation**
   Tkinter is the standard GUI library in Python. Official documentation can be found at Python's library reference: https://docs.python.org/3/library/tkinter.html.
3. **Pygame Library**
   Pygame is a set of Python modules designed for writing video games. Official documentation and resources are available at https://www.pygame.org.
4. **Minimax Algorithm**
   An introduction to the Minimax algorithm and its application in game development: https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory.
5. **Basic Tic Tac Toe Rules**
   General rules and structure of the Tic Tac Toe game: https://en.wikipedia.org/wiki/Tic-tac-toe.
6. **Python Data Structures**
   Guide on lists and dictionaries in Python, essential for board representation: https://docs.python.org/3/tutorial/datastructures.html.
7. **Python Functions and Modular Programming**
   Tutorial on creating reusable functions and organizing code: https://realpython.com/defining-your-own-python-function.
8. **Game Design Principles**
   Overview of principles applied to turn-based games like Tic Tac Toe: https://gamedevelopment.tutsplus.com/articles.
9. **Software Development Life Cycle (SDLC)**
   Explanation of the SDLC models, including the Waterfall model used in this project: https://www.tutorialspoint.com/sdlc/index.htm.
10. **Python Input Validation Techniques**
    Techniques for handling user input, including validation for gaming applications: https://www.w3schools.com/python/python_try_except.asp.