**Networks & Internet course – Final Programming Assignment**

**Tic Tac Toe multiplayer game**

Coded by: Ariel Benillouche. (ID : 209703651)

Coded in: Python (version 3.10), Operating System: Windows 10

Files: 1. TicTacToe\_server.py

2. TicTacToe\_client.py

Sections of this document:

1. Overview
2. Documentation of classes and functions
3. Usage instructions and Illustration examples

I. Overview

The following application is a multiplayer tic tac toe game, that can be played by connecting to a remote server. This application enables multiple clients to play simultaneously and with each other.

The project consists of the following files:

* **TicTacToeServer.py** - Implements the game server, manages game rooms, and handles client communication.
* **TicTacToeClient.py** - Implements the client-side GUI, connects to the server, and allows players to make moves.

II. Documentation of classes and functions:

**Server side (in TicTacToe\_server.py):**

Class: GameRoom

Represents a game room that maintains:

* players: List of connected players.
* board: 3x3 matrix representing the game board.
* current\_turn: Tracks whose turn it is.
* game\_active: Boolean flag indicating if the game is active.

Class: TicTacToeServer

Handles the server operations.

* **\_\_init\_\_(self, host="127.0.0.1", main\_port=3000)**  
  Initializes the server, creates game rooms, and starts listening for client connections.
* **start\_game\_server(self, port)**  
  Starts a game server on a given port and listens for player connections.
* **handle\_client(self, client, port)**  
  Handles a new client connection, assigns them to a game room, and starts the game when two players join.
* **start\_game(self, port)**  
  Begins the game by notifying players and setting the first turn.
* **handle\_game(self, player\_name, client, port)**  
  Main game loop where moves are received, validated, and broadcast to the opponent.
* **broadcast\_move(self, port, row, col)**  
  Sends move updates to both players.
* **broadcast\_reset(self, port)**  
  Resets the game board and notifies players.
* **handle\_disconnect(self, player\_name, port)**  
  Handles player disconnection and resets the game room.

**Client-Side (in TicTacToeClient.py)**

Class: TicTacToeClient

Manages the client-side logic and GUI using Tkinter.

* **\_\_init\_\_(self)**  
  Initializes the client, connects to the server, and sets up the game interface.
* **setup\_gui(self)**  
  Creates the game window and buttons for user interaction.
* **connect\_main\_server(self)**  
  Connects to the main server and retrieves available game rooms.
* **start\_game(self)**  
  Displays available game rooms for the player to join.
* **connect\_game\_server(self, port)**  
  Connects to a selected game room and sends the player’s name.
* **receive\_moves(self)**  
  Listens for updates from the server about opponent moves.
* **handle\_move(self, row, col)**  
  Updates the board with a new move and checks for a winner.
* **on\_click(self, event)**  
  Sends move data to the server when the player clicks on the board.
* **draw\_board(self)**  
  Draws the Tic-Tac-Toe grid.
* **draw\_symbol(self, x, y)**  
  Draws 'X' or 'O' on the board.
* **check\_winner(self)**  
  Checks if there is a winning combination on the board.
* **save\_board(self)**  
  Saves the game state to a CSV file.
* **reset\_game(self)**  
  Clears the board and resets the game state.
* **run(self)**  
  Starts the Tkinter GUI loop.

III. Usage Instructions and Illustration examples

**Usage Instructions:**

Running the Server

1. Open a terminal and navigate to the project folder.
2. Run the server:

python TicTacToeServer.py

1. The server will start and listen for client connections.

Running the Client

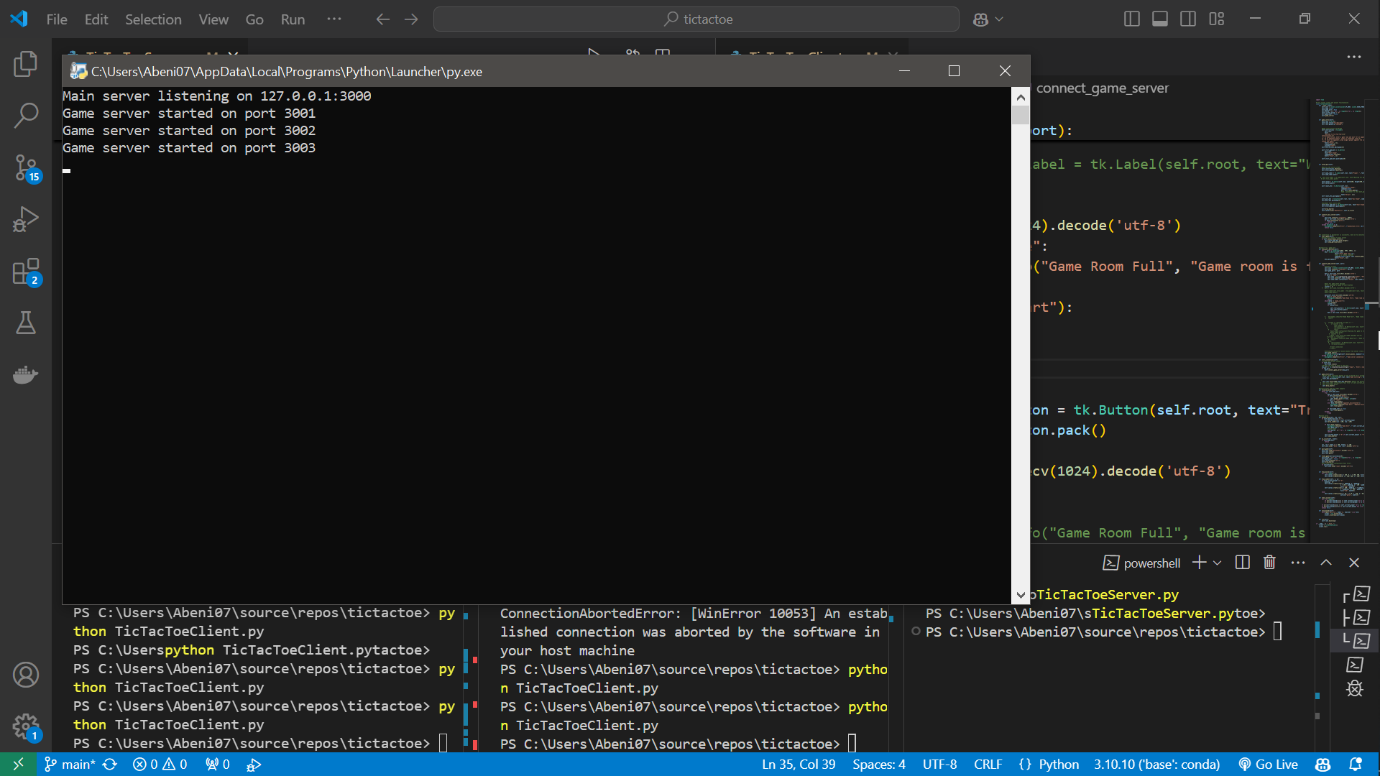
1. Open a terminal and navigate to the project folder.
2. Run the client:

python TicTacToeClient.py

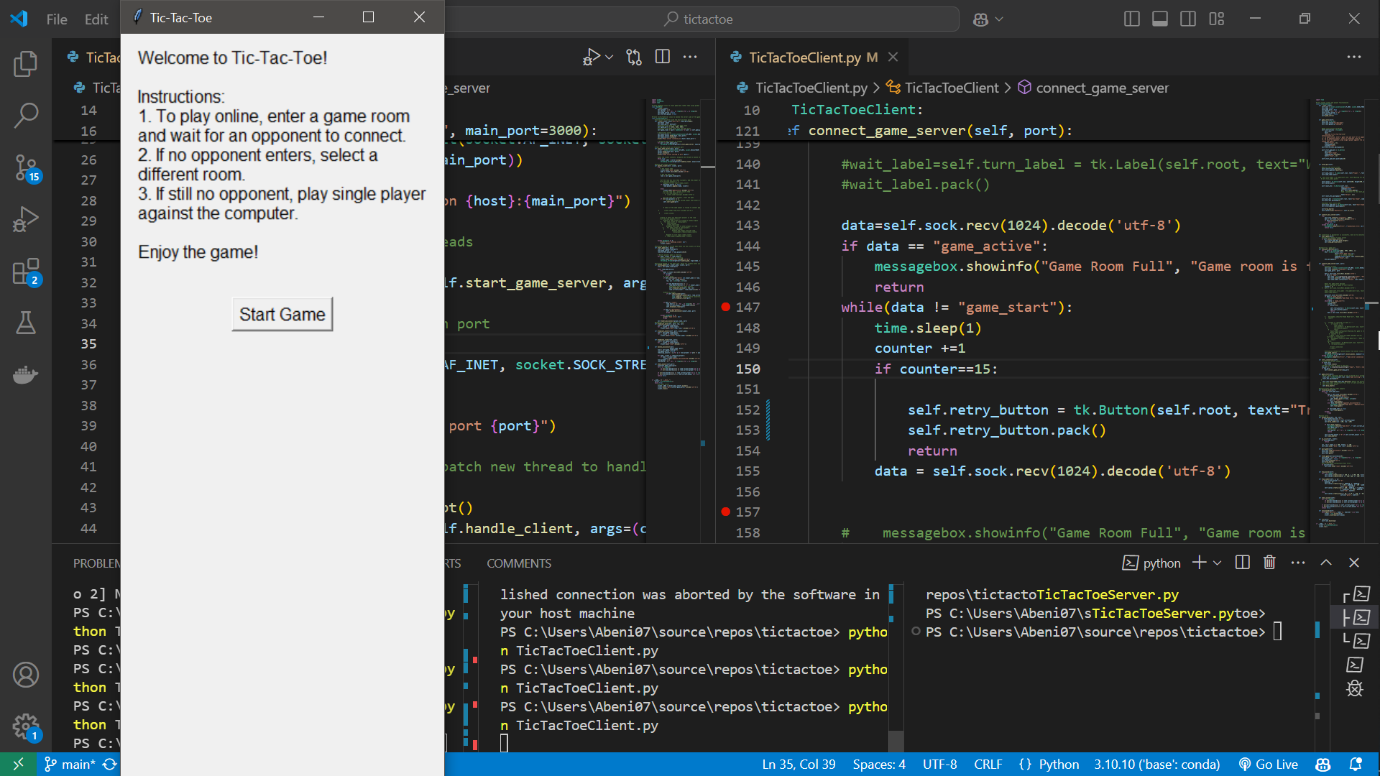
1. Enter your name when prompted.
2. Select a game room.
3. Play the game using the GUI.

**Illustration examples:**

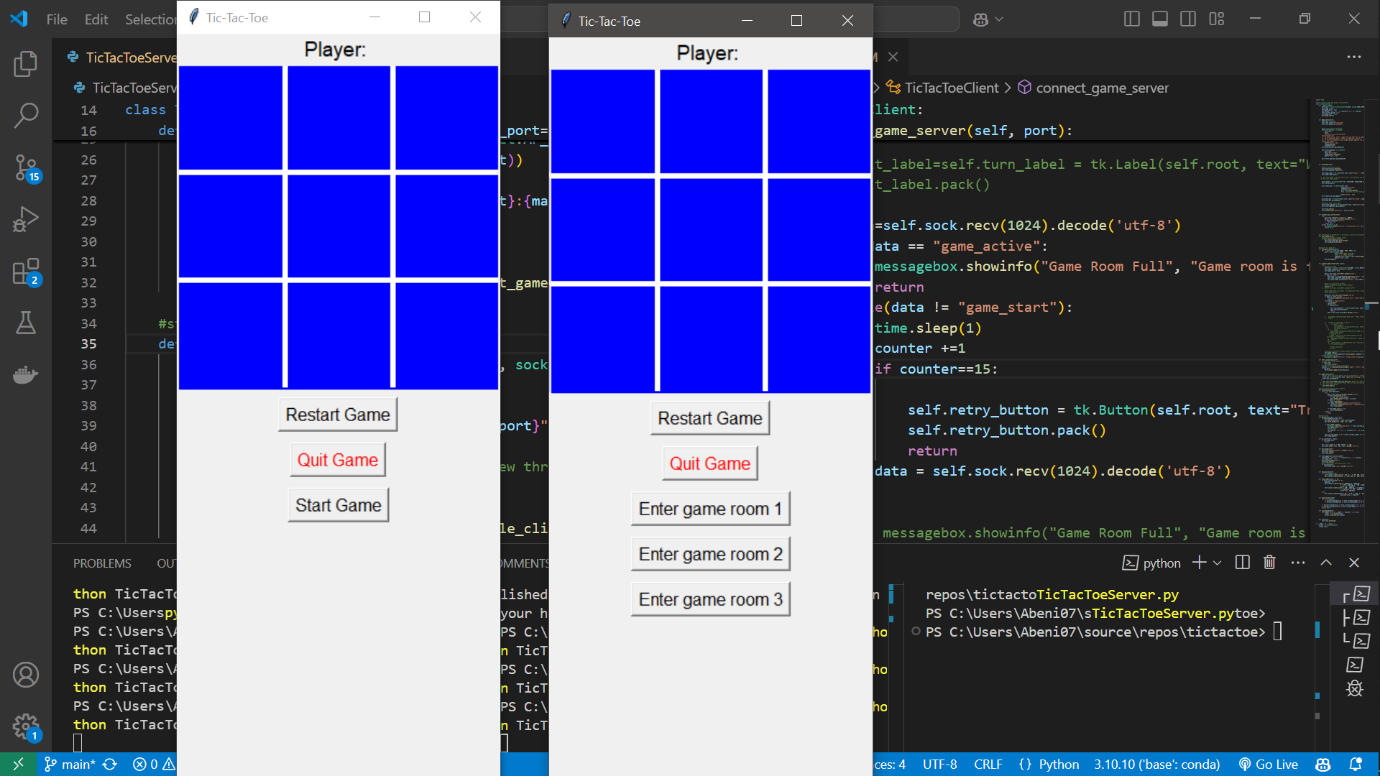
Server running off localhost:3000, and runs sub-servers as threads in ports 3001,3002,3003 to enable multiple games at once.



Running the client. Gui loads with instructions:

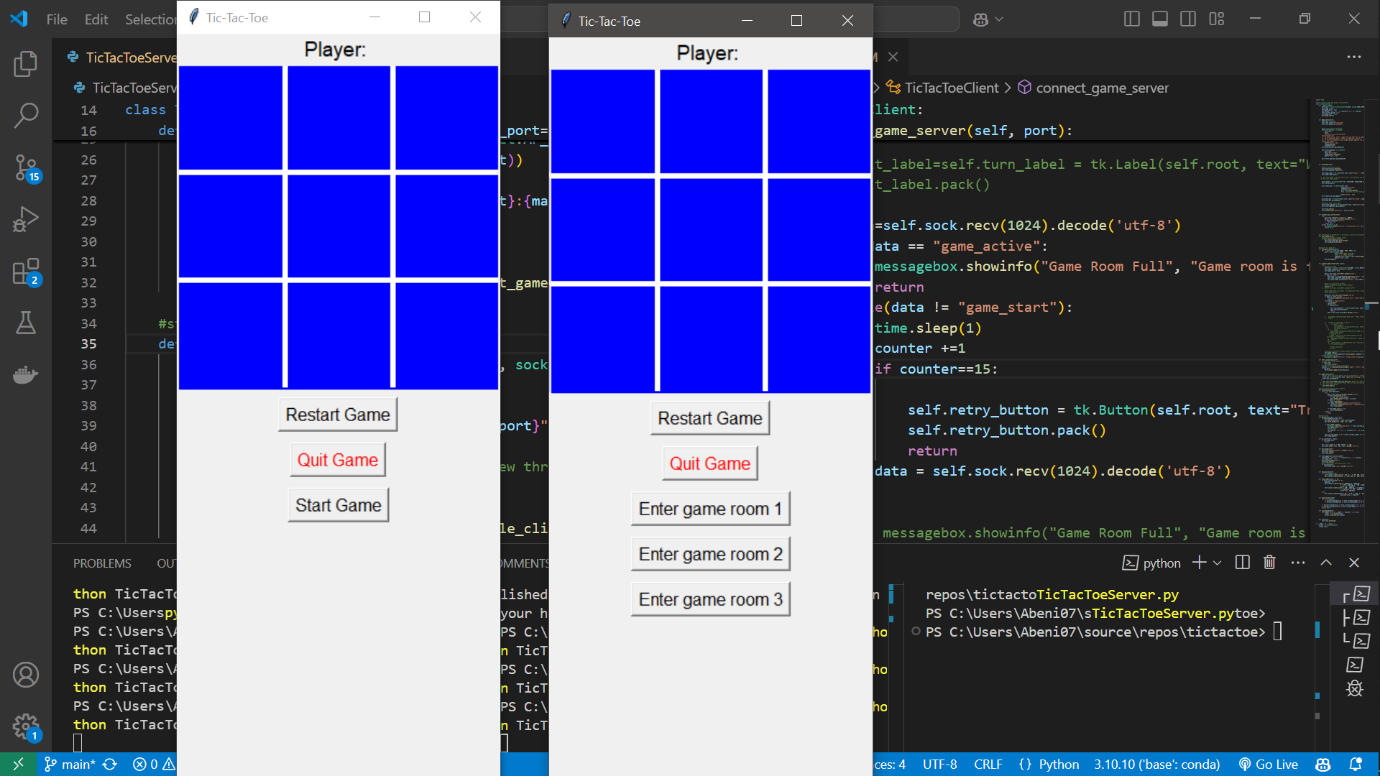


Play screen:



After clicking start game button, select game room (3 game rooms, one per port).

Note: must wait for another player to join the room for game to start.

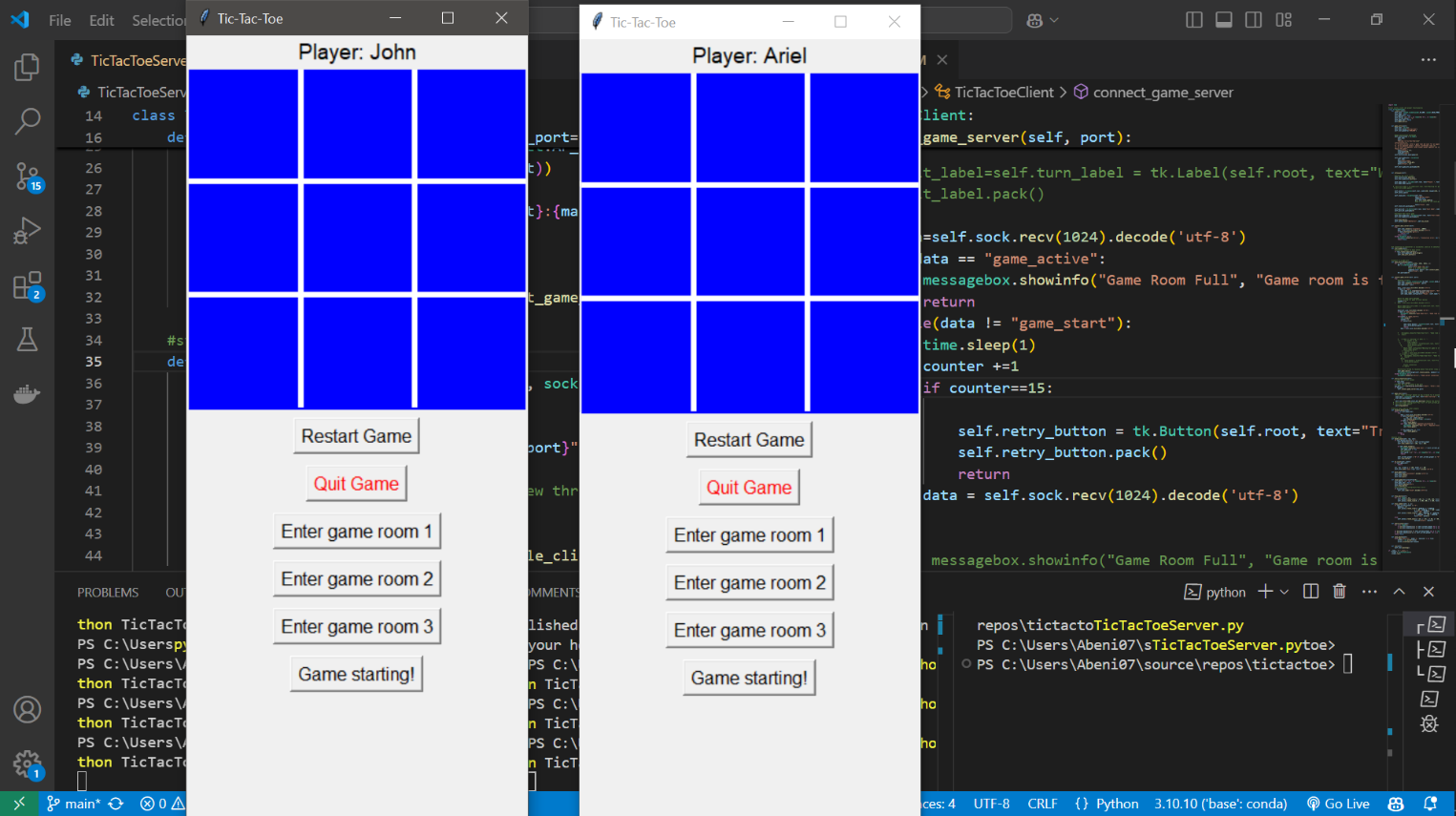


Enter your name:

A screenshot of a computer

AI-generated content may be incorrect.

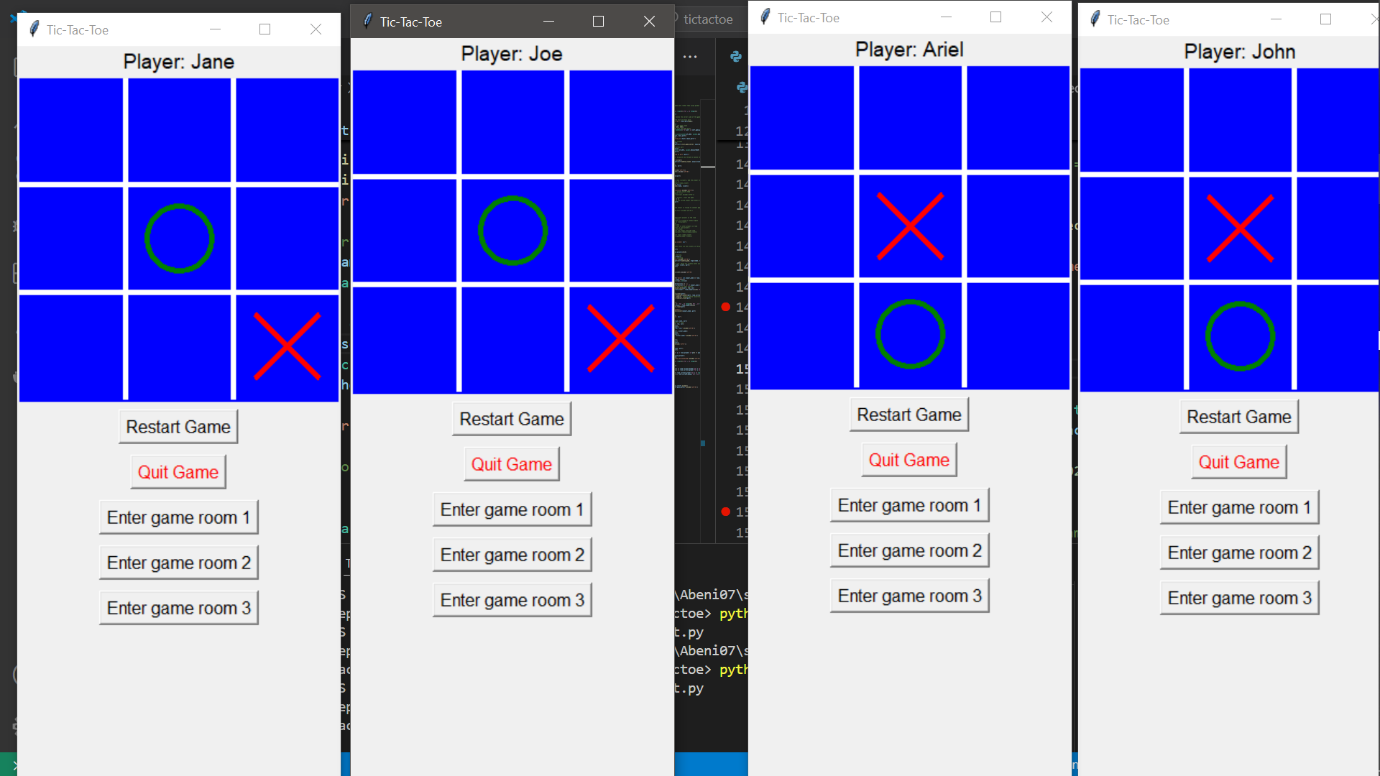
When two players are in room, game will start ("Game starting" button will appear to notify game has started):

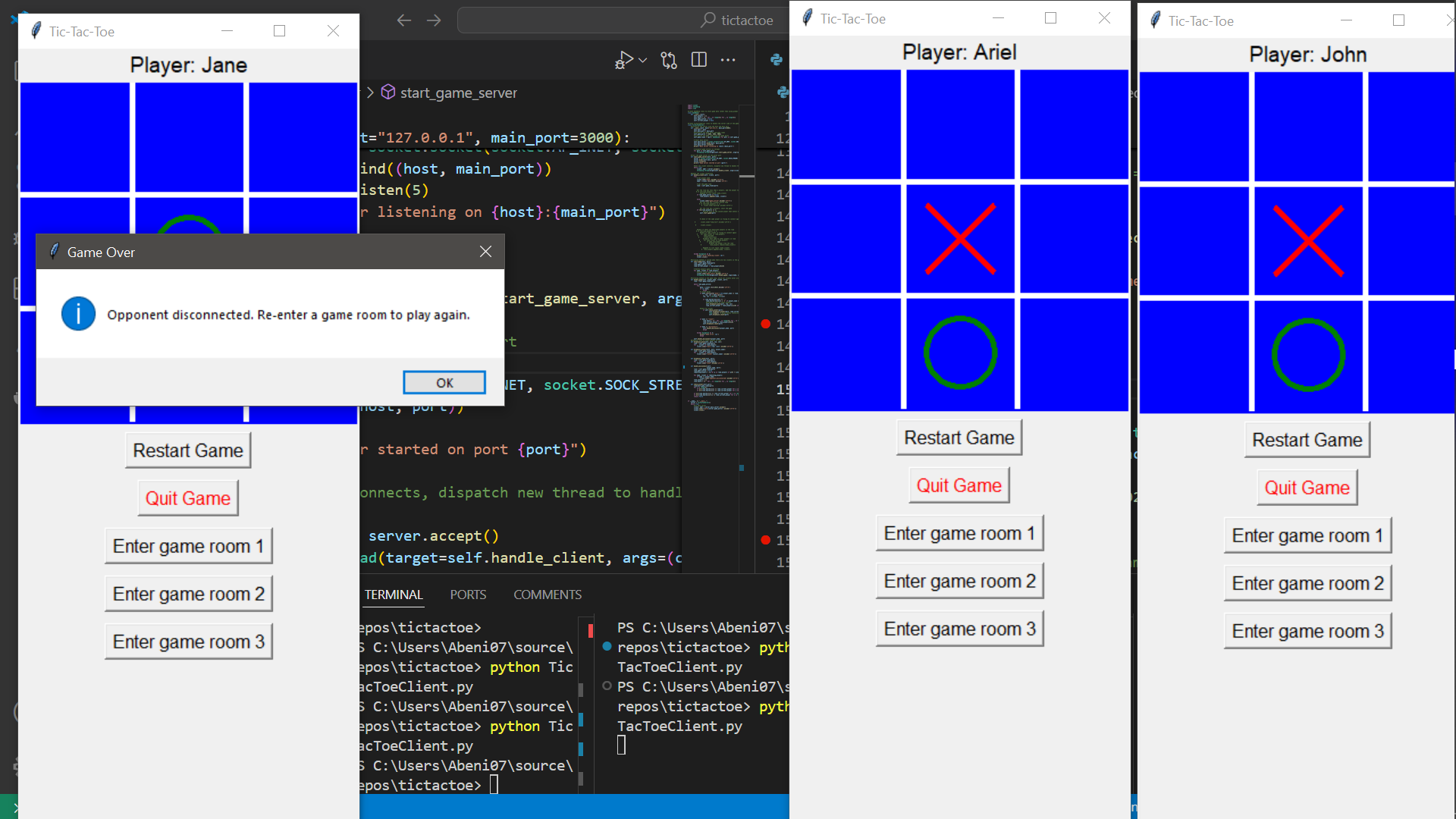


Screens screenshot of a computer

AI-generated content may be incorrect. Game is on. After match ends, new match starts automatically:

Multiple games in parallel:



Notification if player left mid-game: