Tic-Tac-Toe
A Real-Time Multiplayer Game over TCP/IP

CS5700 Computer Networking
Team: Dixuan Zhao, Minyi Zhu, Yuanyuan Wu

Project Goal & Overview

Goal

- Build a real-time multiplayer Tic-Tac-Toe as a TCP/IP networking application

Architecture

- Central server + two clients

Key concepts

- TCP sockets and text-based protocol
- I/O multiplexing with select()
- Multithreaded clients
- Persistent statistics and logging

Extra components

- Tkinter GUI client
- Flask stats dashboard

System Features

Player matching

- Server accepts connections and pairs two users

Turn-based gameplay

- Server enforces turn order and validates moves

Real-time sync & chat

- Both players see the same board; in-game chat

Persistent stats

- Wins / losses / draws stored in stats.json
- Automatically updated after each game

Error handling & disconnections

- Invalid moves → clear error messages
- Player disconnects → opponent wins automatically

Multiple interfaces

- Command-line client + Tkinter GUI client
- Optional Flask dashboard for leaderboard

Architecture & Protocol

Components

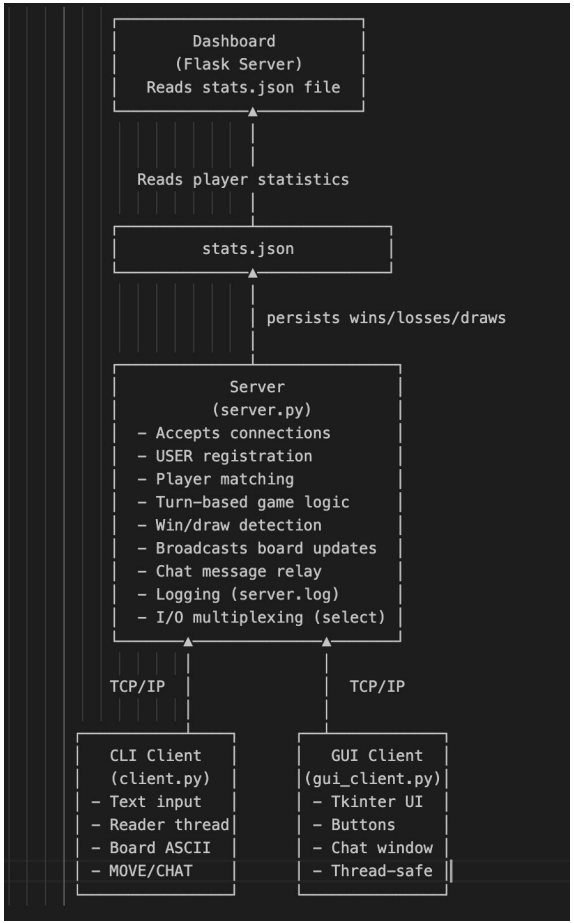
- server.py: centralized game coordinator
- client.py: command-line client
- gui_client.py: Tkinter GUI client
- dashboard.py: Flask stats viewer

Application protocol (line-based)

- Client → Server: USER, MOVE r c, CHAT msg, QUIT
- Server → Client: INFO, BOARD, TURN, START, MSG, STATS, RESULT

Design choice

- Server as single source of truth for board state



Testing & Future Work

Testing

- Localhost: 2 terminals / 2 GUIs vs same server
- Checked:
  - Normal game: win / draw
  - Invalid & out-of-turn moves
  - Chat during gameplay
  - QUIT / disconnect → opponent wins
  - CLI and GUI are compatible

Future Work

- Multiple concurrent games
- Reconnection within a timeout window
- Simple authentication + stronger validation
- Web front end using HTML/JS + WebSockets
- Richer dashboard and analytics

Handoff to Demo

Live Demo of Game

Thank you!