Multiplayer Quiz Game
CENG336 - Computer Networks Project Report

Batuhan Özdemir
Student ID: 210201005

May 18, 2025

Project Description

This project is a multiplayer real-time quiz game implemented in Python using TCP sockets and GUI-based clients built with Tkinter. The server manages multiple clients, broadcasts quiz questions, receives answers, calculates scores, and announces the winner(s).

Technologies Used

- Python 3.x
- socket, threading, json modules
- Tkinter for GUI

System Workflow

1. Server Initialization

The server is started using `python3 server.py`. It begins listening on localhost.

Figure 1: Server started and listening for connections

2. Client Nickname Input

Each client runs `client.py` and is prompted to enter a nickname before joining the game.

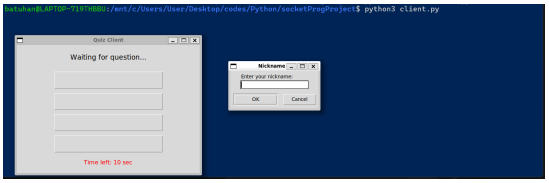

Figure 2: Client entering nickname

3. Server Receives Client Connections

The server logs each connection and assigns the provided nickname.

```
batuhan@LAPTOP-719THBBU:/mnt/c/Users/User/Desktop/codes/Python/socketProgProject$ python3 server.py
[STARTING] Server is starting...
[LISTENING] Server is listening on 127.0.0.1:12345
[NEW CONNECTION] ('127.0.0.1', 38388) connected as a
[NEW CONNECTION] ('127.0.0.1', 48670) connected as b
[NEW CONNECTION] ('127.0.0.1', 35088) connected as c
```

Figure 3: Server logs successful connections

4. Quiz Questions Broadcast

Once the minimum number of players (3) connect, the server broadcasts questions. Each client receives the same question simultaneously with a 10-second timer.

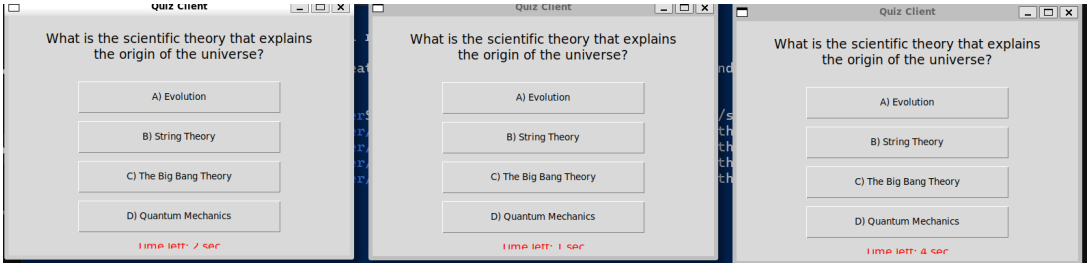

Figure 4: All clients receive the same question

5. Final Scores and Winners

After all players complete the quiz, final scores are sent to each client and also displayed on the server terminal. Players with the highest score are marked as `(Winner)`.

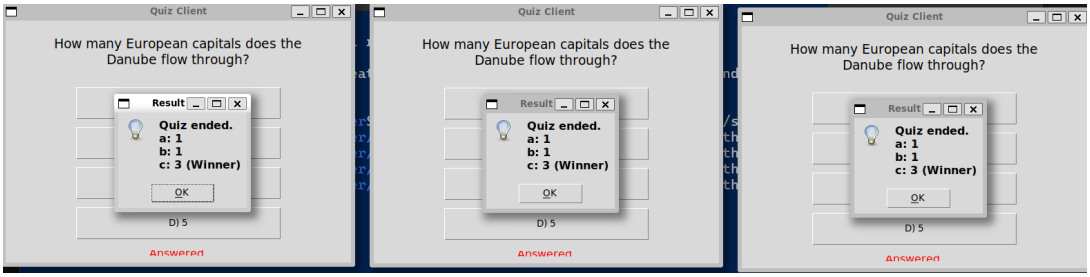

Figure 5: Client view: final results with winner tag

How to Run

1. Run the server:

   ```
   python3 server.py
   ```

2. Run at least 3 clients:

   ```
   python3 client.py
   ```

3. Enter nickname when prompted.
4. Answer questions within 10 seconds per question.
5. Wait until the quiz ends and scores are displayed.

Conclusion

This project demonstrates real-time network programming with concurrency, GUI integration, and synchronization in a multiplayer environment using Python sockets.