CE303 Assignment

The assignment is to build a socket-based client-server system to play a virtual ball. Each client application is a player. Once a player receives a ball, it prompts the user whom to pass the ball to. The clients can connect and disconnect from the server at any time.

The rules of the game

At any point in time, exactly one player has the virtual ball. This player needs to decide who to pass the ball to. (They are allowed to pass the ball to themselves.) Once the decision is made, they pass the ball to the corresponding player.

New players can join the game at any time. The number of players in the game is unlimited. Every player joining the game has to be assigned a unique ID that will not change until they leave the game and will not be reused after they leave the game. All the players including the one with the ball will immediately learn about new players and, hence, the current ball owner can decide to pass the ball to the player who has just joint the game.

Any player can leave the game at any time (the client application can be closed/killed). If the player with the ball leaves the game, the server passes the ball to one of the remaining players.

If there are no players in the game, the server waits until someone joins the game in which case the first player to connect receives the ball.

Client User Interface

Each client has to display their own ID, information about all the players currently in the game and the ID of the player currently having the ball. The client should also be prompting a message to the player that currently has the ball, asking them the ID of the player that they'd like to pass the ball to (e.g. "Whom would you like the ball to? Enter ID").

Server User Interface

The server has to display main events:

- Player joining the game (display the ID of the new player and the list of players currently in the game).
- Player leaving the game (display the ID of that player and the list of players currently in the game).
- The ball being automatically passed to one of the players after the player with the ball left the game (display the ID of the new player with the ball).
- Player passing the ball to another player (display the IDs of both players).

Implementation

Please note that the core requirement of the assignment is to implement the client-server architecture using sockets. The server and each player have to run in separate processes (not thread!). Submissions with a single process running all the players will lose a lot of points. Also, the emphasis should be on reliability. For example, if two players leave at the same time, or a client process is killed (not closed "correctly"), this should be handled correctly.

There has to be server-side validation of data. For example, if player *A* passes the ball to player *B* but player *B* leaves just before that, the server should prevent this, e.g., by returning the ball to *A*.

Only command line interfaces are expected, so there is no need to implement a graphical user interface. You will *not* be awarded extra marks if you implement functionalities that were not required in the assignment (e.g. no extra marks for a GUI).

Submission

The submission should include the code and a report (PDF or Microsoft Word document) of **no more than 1000 words** with the following sections.

Implemented functionality

At the beginning of your report, fill in the table provided in the "Implemented functionality table.docx" document on Moodle.

If your implementation of a function is incomplete, describe what you have done or what is missing in a few words. Otherwise state "yes" or "no". Please note that the implemented functionality table is included in the 1000 words limit.

Protocol

This section should include a detailed description of the application-level protocol. Explain it from the client's point of view: what data does the client send to the server and what data does it expect from the server, and in which order is the data exchanged. Be as specific as possible. This section, together with the assignment brief, should be sufficient for an independent developer to implement a client and a server compatible with your software.

Client Threads

This section should tell what threads are running in the client process. Briefly explain the purpose of each thread and when it is created and terminated.

Server Threads

This section should tell what threads are running in the server process. Briefly explain the purpose of each thread and when it is created and terminated.

Project review

A review of your project – how did it go? Which parts were easy, which parts were challenging? Are there any features you are particularly proud of? Are there any problems with the quality of the program? How was your project management? Is there anything you would do differently next time?

Grading

Attribute	Weight
Implementation	75%
Report	25%

Grading Table for the Implementation Attributes

80-100: The applications implement all the required functionality, function smoothly and are reliable.

70-79: The applications implement all the required functionality, however there might be some minor issues with their reliability, or there could be insignificant issues with the required functionality.

60-69: The application lacks some of the required functionality but an attempt to guarantee reliability is evident from the code.

50-59: The application implements the basic functionality; reliability is not properly considered.

40-49: Some aspects of the basic functionality are implemented.

0-39: The application does not function in a reasonable way, or it does not meet most of the requirements.

Grading Table for the Report Attribute

70-100: The report covers all required sections with the right level of detail at each section.

60-69: The report includes all the required discussions but may lack some details.

50-59: The report may lack some significant details.

40-49: The report includes few details about the protocol and/or threads.

0-39: The report is lacking useful information in most of the sections.

Academic Integrity

You are reminded that this work is for credit towards the composite mark in CE303, and that the work you submit must therefore be your own. Any material you make use of, whether it is from textbooks, the web or any other source (other than the given program code or CE303 lecture / lab worksheets) must be acknowledged in a comment in the program, with the extent of the reference clearly indicated.