ĐỒ ÁN THỰC HÀNH 1 – LẬP TRÌNH SOCKET

Đề 2 – LIVE SCORE

Môn Mạng Máy Tính – 19TN

I. Thông tin của nhóm

- 19120154 – Nguyễn Minh Uyên
- 19120376 – Nguyễn Lê Bảo Thi

II. Đánh giá mức độ hoàn thành

- Mức độ hoàn thành: 97%
- Link đồ án đầy đủ (bao gồm release): https://drive.google.com/file/d/1Bu3EP9aCFlu9I-JDLKvT3vMgVMEIF6qt/view?usp=sharing
- Bảng các việc đã và chưa hoàn thành:

Yêu cầu	Đã hoàn thành	Chưa hoàn thành
KẾT NỐI	• Cho phép client kết nối đến server thông qua kết nối TCP • Cho phép n client kết nối đồng thời đến server (n do quản trị server quyết định) • Cho phép client và server đặt tại các host khác nhau (cho phép client nhập IP của server để kết nối)	
QUẢN TRỊ KẾT NỐI	• Khi client hoặc server mất kết nối đột ngột, không làm chương trình treo hay xảy ra lỗi • Nếu một client mất kết nối không làm ảnh hưởng đến các client khác (Quản trị kết nối đa tiểu trình)	
ĐĂNG NHẬP	Client đăng nhập bằng cách gửi username, password cho server Server nhận thông tin username, password từ client để kiểm tra với thông tin đã lưu trữ tại server	
ĐĂNG KÝ	Client đăng ký bằng cách gửi username, password cho server Server nhận thông tin username, password từ client và kiểm tra với thông	

	tin đã lưu trữ tại server, nếu đã tồn tại, gửi thông báo đến client, yêu cầu đăng ký tài khoản khác	
XEM DANH SÁCH CÁC TRẬN ĐẤU	• Client gửi yêu cầu đến server để nhận về toàn bộ danh sách các trận đấu. • Server sau khi tiếp nhận yêu cầu, gửi về danh sách các trận đấu, client trình bày danh sách các trận đấu kèm tỉ số. • Trả về thông tin trận đấu theo thời gian thực, trận nào đang diễn ra ở phút thứ mấy của trận đấu (trên 90 tổng), trận nào chưa diễn ra ghi giờ trận đấu, trận nào đang nghỉ giữa hiệp (half time - HT), hết trận (full time - FT).	
XEM THÔNG TIN MỘT TRẬN ĐẤU	• Client gửi ID của trận đấu lên server, server tiếp nhận và trả về thông tin chi tiết của trận đấu đó. Client hiển thị các thông tin. • Phía server xây dựng thêm một ứng dụng client dành cho admin, cho phép admin cập nhật thông tin các trận đấu	
QUẢN TRỊ CƠ SỞ DỮ LIỆU	Sử dụng cơ sở dữ liệu có cấu trúc: json	
THOÁT	• Client được phép gửi thông báo ngừng kết nối đến server • Khi đóng server, thông báo ngừng kết nối không hiện lên ngay lập tức trên các client mà khi các client request mới hiện lên thông báo "Server đã ngừng kết nối".	• Server gửi thông báo ngừng kết nối đến tất cả client đang hoạt động ngay khi ngắt kết nối.
GIAO DIỆN	Có thiết kế giao diện đồ họa cho chương trình (GUI) • Client • Server	

III. Kịch bản giao tiếp của chương trình

- Giao thức trao đổi giữa client và server: TCP

Ta khởi tạo socket với kiểu thiết lập kết nối là Ipv4 và giao thức là TCP.

- Các vấn đề kết nối:
  - Kết nối: Khi mở server, server sẽ được người quản trị nhập số client được kết nối tối đa cùng một lúc tới server. Sau đó server được mở ra tạo một socket với địa chỉ là địa chỉ của máy đặt server và port 65432 để lắng nghe kết nối từ tối đa n client. Lúc này server chuyển sang chờ đợi kết nối từ client trong 1 thread khác (thread chính ban đầu dùng để chờ tín hiệu ngừng kết nối). Client sẽ nhập IP của server để kết nối tới server. Mỗi khi nhận được yêu cầu kết nối server sử dụng một luồng mới để giao tiếp với client này. Client được chấp nhận và được thêm vào danh sách các clients kết nối trên server và bắt đầu quá trình giao tiếp với server để thực hiện chương trình.
  - Quản trị kết nối: Khi server bị mất kết nối đột ngột thì khi các client gửi yêu cầu tới server sẽ nhận được thông báo “Server đã ngừng kết nối”, người dùng cần bấm nút đóng giao diện client để kết thúc chương trình. Khi client mất kết nối thì sẽ gửi thông điệp “QUIT” tới server, server sẽ đóng kết nối với client đó, hiển thị thông tin client đó đã ngừng kết nối và kết thúc tiểu trình giao tiếp với client đó. Vì vậy chương trình sẽ không bị treo và sẽ không ảnh hướng tới các client đang kết nối khác.
  - Thoát kết nối: Khi một client đã được đáp ứng các yêu cầu của mình và không có công việc để tiếp tục thì sẽ đóng cửa sổ và sẽ gửi thông điệp “QUIT” tới server để đóng kết nối tương tự như trường hợp mất kết nối. Khi server muốn đóng kết nối sẽ đóng các socket đang giao tiếp với tất cả các client làm cho client mất kết nối. Sau đó, khi client gửi yêu cầu tới server sẽ được hiển thị thông báo “Server đã ngừng kết nối” và người dùng sẽ đóng cửa sổ để kết thúc chương trình.
- Cấu trúc thông điệp: Client gửi yêu cầu cho server bằng các thông điệp ngắn gọn sau:
  - “LOGIN”: yêu cầu thực hiện đăng nhập. Client gửi thêm username, password được format kiểu dictionary: {‘usr’: <username>, ‘psw’: <password>}. Nếu đăng nhập admin thành công, server gửi lại thông điệp “2”. Nếu đăng nhập user thành công, server gửi “1”. Nếu sai mật khẩu server gửi “-1”. Nếu tài khoản không tồn tại server gửi “0”.
  - “REGIST”: yêu cầu thực hiện đăng kí. Client gửi thêm username, password được format kiểu dictionary: {‘usr’: <username>, ‘psw’:

<password>} . Nếu đăng kí không thành công, server gửi lại "0", ngược lại gửi "1".

- "ADDMT": admin yêu cầu thêm trận đấu. Admin gửi thêm thông tin chung của trận đấu kiểu dictionary theo format: {'id': <ID trận đấu>, 'team1': <tên đội 1>, 'team2': <tên đội 2>, 'time': <thời gian bắt đầu theo format yyyy-mm-dd hh:mm>, 'score': <tỉ số>}. Nếu thêm thành công, server gửi "1", ngược lại gửi "0".
- "UPDMT": admin yêu cầu cập nhật thời gian và tỉ số trận đấu. Admin gửi thêm thông tin cập nhật kiểu dictionary theo format: {'id': <ID trận đấu cần cập nhật>, 'time': <thời gian bắt đầu theo format yyyy-mm-dd hh:mm cập nhật>, 'score': <tỉ số cập nhật>}. Nếu cập nhật thành công, server gửi "1", ngược lại gửi "0".
- "UPDEV": admin yêu cầu thêm sự kiện trận đấu. Admin gửi thêm thông tin về sự kiện cần thêm kiểu dictionary theo format: {'id': <ID trận đấu>, "events":{'time': <số phút hiện tại hoặc HT, FT>, 'type': <loại sự kiện (bàn thắng - goal, bàn thắng bằng penalty – goalPen, thẻ vàng - YellowCard, thẻ đỏ - RedCard)>, 'team': <tên đội làm nên sự kiện>, 'player': <tên cầu thủ làm nên sự kiện>, 'assist': <tên cầu thủ kiến tạo (nếu sự kiện là bàn thắng, nếu không thì để trống), 'score': <tỉ số mới (trong trường hợp có bàn thắng hoặc HT, FT, không thì để trống)>}}. Nếu thêm thành công, server gửi "1", ngược lại gửi "0".
- "LISTMT": user yêu cầu xem toàn bộ danh sách các trận đấu. Server gửi toàn bộ danh sách kiểu dictionary theo format như mục "Cách tổ chức cơ sở dữ liệu" bên dưới. Client nhận và cho hiển thị danh sách các trận đấu đã kết thúc kèm tỉ số, các trận đấu chưa kết thúc hoặc chưa bắt đầu chỉ để thời gian bắt đầu trận đấu.
- "LISTMRT": user yêu cầu xem danh sách các trận đấu theo thời gian thực. Server cũng gửi toàn bộ danh sách như trên. Client nhận và cho hiển thị các trận đấu cùng tỉ số và số phút hiện tại trong trận nếu trận đang diễn ra. Nếu trận chưa diễn ra để thời gian bắt đầu và tỉ số 0-0.
- "MTCDT": user yêu cầu xem chi tiết trận đấu, gồm các sự kiện chính trong trận. User gửi kèm ID trận đấu. Client tìm trận đấu theo ID nhận được và gửi lại các sự kiện của trận đấu đó theo format dictionary: {"events":[{'time': <số phút hiện tại hoặc HT, FT>, 'type': <loại sự kiện (bàn thắng - goal, bàn thắng bằng penalty – goalPen, thẻ vàng - YellowCard, thẻ đỏ - RedCard)>, 'team': <tên đội làm nên sự kiện>, 'player': <tên cầu thủ làm nên sự kiện>, 'assist': <tên cầu thủ kiến tạo

(nếu sự kiện là bàn thắng, nếu không thì để trống), ‘score’: <tỉ số mới (trong trường hợp có bàn thắng hoặc HT, FT, không thì để trống)>}]}. Nếu không tìm thấy ID trận đấu, server gửi “0”.
    - “QUIT”: client gửi thông báo thoát (ngừng kết nối).
- Kiểu dữ liệu của thông điệp:
    - Yêu cầu từ client và tín hiệu gửi về từ server: Yêu cầu từ client và tín hiệu gửi về từ server đều được nhận và gửi bởi hàm sendData và receive. Hàm sendData sử dụng hàm .sendall để gửi dữ liệu theo kiểu bytes đã được mã hóa theo kiểu “utf-8”. Ở bên còn lại khi nhận dữ liệu bằng hàm receive dùng hàm .recv giá trị trả về là bytes đại diện cho dữ liệu đã nhận được. Lượng dữ liệu tối đa nhận được cùng một lúc là BUFF_SIZE = 1024 trừ khi socket bị lỗi. Sau khi nhận dữ liệu xong dữ liệu sẽ được giải mã và xóa kí tự trắng dư thừa đầu và cuối dữ liệu đó là chuỗi python thông thường và dùng nó để nhận yêu cầu để thực hiện hay là nhận thông báo tín hiệu để biết kết quả của yêu cầu của mình.
    - Data dữ liệu nhận và gửi: Data nhận và gửi về từ client và server liên quan đến dữ liệu của LIVE SCORE đều được nhận và gửi dưới dạng .json. Trước khi gửi dữ liệu ta sẽ dùng lệnh json.load() để lấy dữ liệu từ file dữ liệu và sử dụng json.dumps() để ghi dữ liệu json vào chuỗi python và gửi đi bằng hàm sendData tương tự như yêu cầu từ client và tín hiệu gửi về từ server. Bên nhận nhận được dữ liệu qua hàm receive tương tự như yêu cầu từ client và tín hiệu gửi về từ server và dùng hàm json.loads() để giải mã dữ liệu từ chuỗi python thành dữ liệu json để thao tác và ghi vào file dữ liệu cần thiết bằng json.dump().
- Cách tổ chức cơ sở dữ liệu: Cơ sở dữ liệu được tổ chức theo cấu trúc dictionary và lưu trong các file .json, chia thành 2 file:
    - “account.json”: để lưu các tài khoản đã được đăng ký gồm username và password (bao gồm cả tài khoản admin). Cấu trúc dữ liệu: { ‘account’: [{‘usr’: <username>, ‘psw’: <password>}] }
    - “matchDetail.json”: để lưu lại danh sách các trận đấu gồm ID trận đấu, thời gian, đội thứ nhất, đội thứ hai, tỉ số. và sự kiện xảy ra trong trận đấu đó gồm thời gian, loại sự kiện, đội, cầu thủ (người kiến tạo), tỉ số. Cấu trúc dữ liệu: { ‘match’: [{‘id’: <ID trận đấu>, ‘team1’: <tên đội 1>, ‘team2’: <tên đội 2>, ‘time’: <thời gian bắt đầu theo format yyyy-mm-dd hh:mm>, ‘score’: <tỉ số>, “events”:[{‘time’: <số phút hiện tại hoặc HT, FT>, ‘type’: <loại sự kiện (bàn thắng - goal, bàn thắng bằng penalty – goalPen, thẻ vàng - YellowCard, thẻ đỏ - RedCard)>, ‘team’: <tên đội

làm nên sự kiện>, ‘player’: <tên cầu thủ làm nên sự kiện>, ‘assist’: <tên cầu thủ kiến tạo (nếu sự kiện là bàn thắng, nếu không thì để trống), ‘score’: <tỉ số mới (trong trường hợp có bàn thắng hoặc HT, FT, không thì để trống)>}]}]}.

IV. Môi trường lập trình và các framework hỗ trợ để thực thi ứng dụng

- Môi trường lập trình:
  - Hệ điều hành Windows 10.
  - Ngôn ngữ lập trình: Python 3.
- Các thư viện hỗ trợ để thực thi ứng dụng:
  - socket: Hỗ trợ tạo object socket với các phương thức căn bản (listen, bind, sendall, recv, close)
  - _thread: Hỗ trợ lập trình đa luồng (cấp thấp hơn)
  - Threading: Hỗ trợ lập trình đa luồng (cấp cao hơn)
  - Tkinter: Hỗ trợ cài đặt, thiết kế giao diện
  - Pillow: Hỗ trợ xử lý ảnh
  - Datetime: Hỗ trợ xử lý thời gian kết hợp với ngày
  - Timedelta: Hỗ trợ cộng trừ datetime
  - Time: Hỗ trợ xử lý thời gian độc lập trong ngày
  - Queue: Hỗ trợ hàng đợi đa người dùng (Chia sẻ dữ liệu giữa các luồng)
  - Functools: Hỗ trợ các hàm bậc cao hoạt động trên các chức năng khác
  - Json: Hỗ trợ tổ chức cơ sở dữ liệu có cấu trúc .json
  - Logging: Hỗ trợ cấu hình, phân chia level thông báo lỗi lên console. (Hỗ trợ hiển thị thao tác console lên serverGUI)
  - Signal: Cung cấp các cơ chế đặc biệt để communicate giữa các process.

V. Hướng dẫn sử dụng các tính năng của chương trình

Sau khi đã cài đặt các thư viện cần cho chương trình được liệt kê ở mục IV, chạy file serverGUI.py bằng lệnh python serverGUI.py để khởi động chương trình server. Tại cửa sổ đầu tiên, nhập số lượng client cho phép kết nối đồng thời đến server rồi bấm Submit. Giao diện chính của server sẽ hiện lên với tên chương trình “Live Score” và các chú thích hiển thị bên trái, phần còn lại là cửa sổ chính - nơi hiển thị các thông báo kết nối và giao tiếp với các client.

Với mỗi client, khởi động bằng cách chạy file client.py. Cửa sổ đầu tiên yêu cầu client nhập IP của server cần kết nối và click Connect to server để gửi yêu cầu kết nối. Nếu kết nối không thành công (do server chưa mở,…) sẽ hiện thông báo chưa kết nối được, ngược lại sẽ hiện thông báo “Kết nối đến server thành công”.

Sau khi 1 client kết nối đến server thành công, màn hình server hiện thông báo client thứ mấy đã kết nối, địa chỉ IP và port của client đó. Phía client hiển thị cửa sổ đăng nhập, đăng kí. Client có thể đăng nhập (thầy có thể đăng nhập bằng những tài khoản có sẵn như user: thi, pass: 123 hoặc user: uyen, pass: 789) hoặc đăng kí nếu chưa có tài khoản. Sau khi đăng kí, chương trình sẽ trở lại cửa sổ đăng nhập. Với admin, đăng nhập bằng user: admin, pass: adm để sử dụng quyền admin là cập nhật thông tin trận đấu.

Sau khi client đăng nhập với vai trò user, chương trình chuyển sang cửa sổ "Danh sách trận đấu" với 3 chức năng:

- Xem toàn bộ danh sách trận đấu: bấm nút "List match", toàn bộ danh sách trận đấu trong data hiện lên màn hình với các thông tin: ID trận, thời gian bắt đầu (FT nếu trận đấu đã kết thúc), tên 2 đội, tỉ số (nếu trận đấu đã kết thúc).
- Xem danh sách trận đấu ở thời gian thực: bấm nút "List match real time", danh sách các trận đấu hiện lên màn hình với các thông tin như chức năng List match nhưng ở mục thời gian là số phút hiện tại của trận đấu nếu trận đấu hiện đang diễn ra, HT: nghỉ giữa trận, FT: trận đã kết thúc, hoặc thời gian trận đấu bắt đầu nếu trận đấu chưa diễn ra. Mục tỉ số là tỉ số được cập nhật theo thời gian thực.
- Xem thông tin chi tiết 1 trận đấu: ở ô "Match detail", nhập ID trận đấu cần xem và bấm nút "Xem" (thầy có thể nhập ID "1234" để xem chi tiết trận đấu trong data có sẵn). Trên cửa sổ mới hiện ra, bấm nút "Hiển thị danh sách sự kiện" để xem chi tiết các sự kiện của trận đấu. Khi xem xong bấm nút đóng cửa sổ để quay lại cửa sổ chính của chương trình, có thể nhập lại ID trận đấu khác để xem tiếp.
- Bấm nút đóng cửa sổ chính để thoát chương trình và báo cho server biết client đã ngưng kết nối.

Với admin, sau khi đăng nhập, chương trình chuyển sang cửa sổ làm việc chính của admin với 3 chức năng:

- Thêm trận đấu: bấm nút "Thêm trận". Trên cửa sổ "Thêm trận đấu", nhập ID trận (không được trùng với ID các trận đã có, nếu trùng sẽ bị báo lỗi), nhập thời gian diễn ra trận đấu theo format "yyyy-mm-dd hh:mm" (hour tính từ 0 đến 23), nhập tên 2 đội và tỉ số (nhập "0-0" nếu trận chưa bắt đầu), sau đó bấm nút "Add match" để thêm trận đấu vào file chứa data. Bấm nút đóng cửa sổ khi không thêm nữa.

- Cập nhật thời gian và tỉ số: nhập ID trận đấu cần cập nhật, nhập thời gian bắt đầu trận đấu và tỉ số mới theo định dạng như phần "Thêm trận đấu" (cập nhật cả thời gian và trận đấu cùng lúc, nếu có 1 phần không muốn thay đổi thì nhập lại như cũ). Bấm "Update match" để cập nhật vào file chứa data. Đóng cửa sổ khi không cập nhật nữa.
- Thêm sự kiện cho trận đấu: nhập ID trận đấu cần thêm (nếu ID trận đấu không tồn tại sẽ báo lỗi); nhập thời gian là số phút trong 1 trận đấu (0 tới 90), HT: giữa trận, hoặc FT: cuối trận; chọn loại sự kiện là bàn thắng (goal), bàn thắng phạt đền (goalPen), thẻ vàng (YellowCard) hay thẻ đỏ (RedCard); nếu loại sự kiện là bàn thắng, nhập cầu thủ ghi bàn (player) và cầu thủ kiến tạo (assist), các loại sự kiện còn lại nhập cầu thủ đá phạt đền hoặc cầu thủ bị phạt thẻ; nhập tỉ số nếu sự kiện làm thay đổi tỉ số (tỉ số mới sẽ được cập nhật đồng thời lên mục tỉ số ở phần thông tin chung). Nếu thời gian là HT hoặc FT thì bỏ qua các mục loại sự kiện và tên cầu thủ. Bấm nút "Update Event" để gửi yêu cầu. Đóng cửa sổ khi đã thêm xong.
- Đóng cửa sổ chính để thoát chương trình và gửi thông báo ngắt kết nối cho server.

Ngừng kết nối của server với các client bằng cách đóng cửa sổ chính của server. Nếu vẫn còn client chưa thoát chương trình thì nếu client đó gửi request cho server sẽ hiện lên thông báo "Server đã ngừng kết nối". Khi đó client đóng các cửa sổ của mình để kết thúc chương trình.

VI. Bảng phân công công việc

	Nguyễn Minh Uyên	**Nguyễn Lê Bảo Thi**

Công việc	– Kết nối
	– Quản lý kết nối
	– Đăng nhập
	– Đăng ký
	– Xem thông tin 1 trận đấu:
	  Gửi, nhận thông tin chi tiết trận đấu.
	  Admin đăng nhập, thêm trận, cập nhật tỉ số, thời gian và sự kiện trong trận.
	- Thoát:
	• Server gửi thông báo ngừng kết nối.
	• Client gửi thông báo ngừng kết nối.
		– Giao diện đồ họa Client
		– Giao diện đồ họa Server
		– Xem danh sách các trận đấu:
		• Xem toàn bộ danh sách
		• Xem danh sách trận đấu theo thời gian thực
		– Xem thông tin 1 trận đấu:
		• Client hiển thị các thông tin.
		– Quản lý cơ sở dữ liệu: sử dụng cơ sở dữ liệu có cấu trúc json.

VII. Tài liệu tham khảo

- Giáo trình mạng máy tính, slides mạng máy tính
- Stack Oveflow
- Geeksforgeeks
- Tkinter tutorial
- Real Python
- docs.python.org
- Beenje.github.io (blog logging to a tkinter scrolledText widget)
- w3schools.com