I. Miêu tả ứng dụng

Tên đề tài: *Xây dựng ứng dụng Chat Client–Server bằng ngôn ngữ C (TCP Socket)*

Mô tả ngắn gọn:
Ứng dụng chat hai chiều cho phép nhiều người dùng đăng nhập và trò chuyện theo mô hình Client–Server.
Người dùng có thể gửi tin nhắn cá nhân, tạo nhóm chat, kết bạn, lưu tin nhắn offline và xem lịch sử.

Các chức năng chính:

- Đăng ký / đăng nhập người dùng
- Quản lý kết nối nhiều client
- Chat 1–1
    - Khi có 1 trong 2 user ngắt kết nối hoặc yêu cầu dừng cuộc trò chuyện, thông báo cho bên còn lại biết.
    - Cho phép user chuyển sang cuộc trò chuyện khác (với user khác nếu cần)
- Chat nhóm
    - User tạo nhóm chat
    - User gửi yêu cầu tham gia nhóm tới các user khác
    - Tham gia và rời nhóm chat
    - Gửi thông điệp tới các nhóm
- Quản lý bạn bè (thêm, chấp nhận, hủy)
- Ghi log, lưu offline message
- Giao diện CLI/GUI đơn giản
- Các chức năng mở rộng
    - Gửi/nhận file nhỏ (file .txt hay .jpg nhỏ)

Cách thức sử dụng (user flow):

1. Server được khởi động trước → lắng nghe cổng TCP.
2. Client khởi động, nhập IP server → gửi yêu cầu đăng nhập.
3. Khi đăng nhập thành công:
    - Người dùng thấy danh sách bạn bè, nhóm chat.
    - Có thể chọn gửi tin nhắn 1–1 hoặc chat nhóm.
4. Khi offline, tin nhắn sẽ được lưu lại và gửi khi đăng nhập lại.
5. Server lưu log hoạt động và quản lý các kết nối.

II. CƠ SỞ LÝ THUYẾT

1. Mô hình Client-Server

- **Nguyên tắc:** Ứng dụng hoạt động theo mô hình tập trung, trong đó **Client** là bên yêu cầu dịch vụ và **Server** là bên cung cấp và quản lý tài nguyên.
- **Ưu điểm:** Quản lý tập trung dữ liệu (tài khoản, tin nhắn offline), dễ dàng mở rộng và duy trì trạng thái người dùng (online/offline).

2. Giao thức TCP (Transmission Control Protocol)

- **Nguyên tắc:** Sử dụng TCP cho việc truyền tải dữ liệu. TCP là giao thức **hướng kết nối (connection-oriented)**, đảm bảo dữ liệu được truyền đi **đáng tin cậy (reliable)**, **theo thứ tự (in-order)** và **không bị mất mát**.
- **Ứng dụng:** Lý tưởng cho ứng dụng chat vì yêu cầu độ tin cậy cao cho tin nhắn, đăng nhập, và truyền file.

3. Kỹ thuật Đa luồng (Multi-threading)

- **Server-side:** Server sử dụng kiến trúc đa luồng. Khi một Client mới kết nối, Server sẽ tạo ra một **luồng (thread)** riêng biệt để xử lý tất cả các yêu cầu từ Client đó.
- **Mục đích:** Đảm bảo Server có thể xử lý đồng thời nhiều kết nối Client mà không làm gián đoạn các Client khác (tính **Concurrency**).

Protocol tự định nghĩa: message có dạng
TYPE|FROM:duc|TO:nhi|CONTENT:Hello|TIME:2025-10-28 10:00

- **Các loại message: LOGIN, REGISTER, MSG, GROUP_MSG, FRIEND_REQ, OFFLINE_SYNC, …**
- **File I/O: Lưu thông tin người dùng, log và tin nhắn offline (fprintf, fscanf, fopen, …)**
- **Đồng bộ dữ liệu client–server khi người dùng reconnect.**
- **Ngôn ngữ lập trình: C**

Thành viên 1 (Server-side) – 14 điểm - Trịnh Minh Đức

STT	Đề mục	Mô tả Chức năng	Các hàm chính	Độ ưu tiên

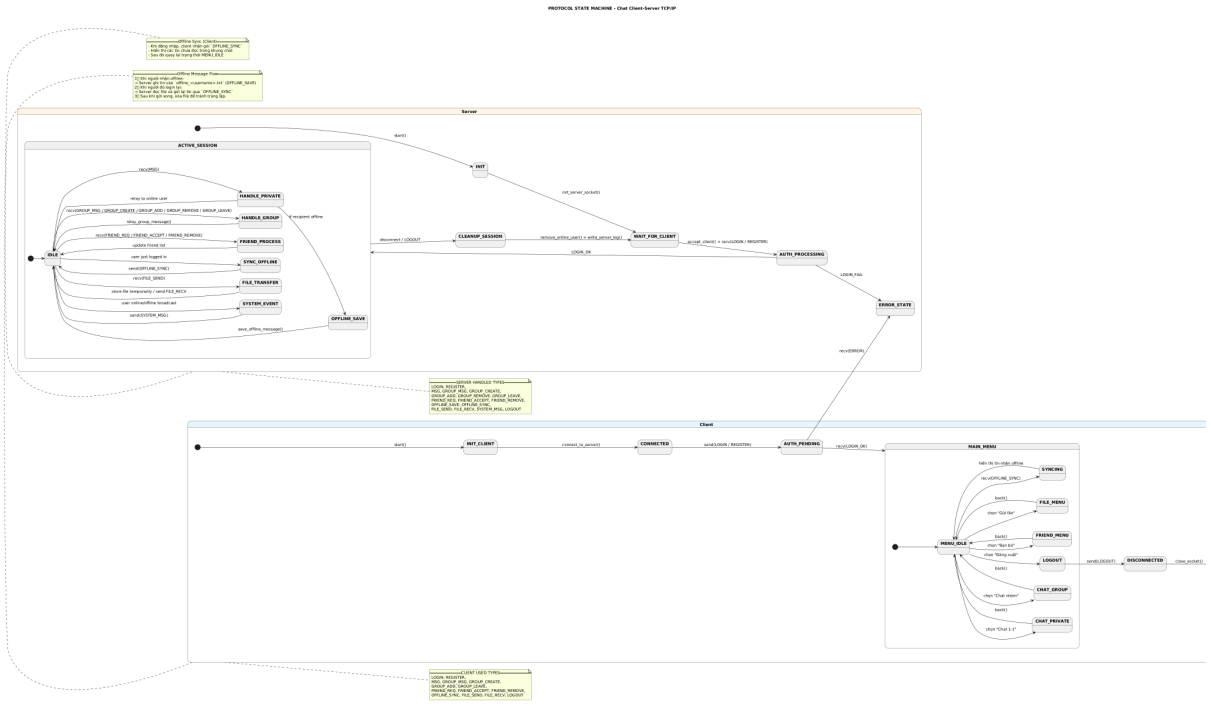
1	**Khởi tạo Socket TCP**	Tạo socket, bind, lắng nghe kết nối, chấp nhận client mới	`init_server_socket()`, `accept_client()`	2
2	**Xử lý Luồng TCP**	Nhận/gửi gói tin, xử lý chống phân mảnh dữ liệu lớn	`recv_message()`, `send_message()`	1
3	**Đa Luồng**	Phân bổ một thread riêng biệt cho mỗi Client kết nối	`pthread_create()`, `client_thread()`	2
4	**Quản lý Tài khoản**	Xử lý yêu cầu Đăng ký / Đăng nhập của người dùng	`handle_register()`, `handle_login()`	2
5	**Danh sách Online**	Theo dõi trạng thái online/offline của tất cả người dùng	`add_online_user()`, `remove_online_user()`	1
6	**Chat 1-1**	Chuyển tiếp tin nhắn riêng tư giữa hai người dùng	`relay_private_message()`	2
7	**Chat Nhóm (Backend)**	Tạo nhóm, **quản lý thành viên** và chuyển tiếp tin nhắn trong nhóm	`create_group()`, `relay_group_message()`	2
8	**Offline Message**	Lưu trữ và gửi lại tin nhắn chưa đọc khi người dùng đăng nhập	`save_offline_message()`, `send_offline_messages()`	1
9	**Xử lý File Nhỏ**	**Nhận, lưu tạm, và chuyển tiếp** các file định dạng text/image nhỏ	`handle_file_transfer()`, `relay_file()`	**1**

Thành viên 2 (Client-side) – 14 điểm - Hoàng Yến Nhi

STT	Đề mục	Mô tả Chức năng	Các hàm chính	Độ ưu tiên
1	**Kết nối TCP**	Tạo socket và thiết lập kết nối tới Server (IP/PORT)	connect_to_server()	1
2	**Giao diện (CLI/GUI)**	Thiết kế menu chức năng, khung chat và hiển thị tin	show_menu(), display_chat()	2
3	**Đăng nhập / Đăng ký**	Gửi gói tin yêu cầu LOGIN, REGISTER tới Server	login_screen(), register_screen()	1
4	**Chat 1-1**	Gửi và nhận PRIVATE_MESSAGE theo thời gian thực	chat_private()	2
5	**Quản lý Nhóm**	Tạo nhóm mới, tham gia, hoặc rời khỏi nhóm hiện tại **(gửi yêu cầu)**	group_menu(), send_group_command()	2
6	**Chat Nhóm**	Hiển thị tin nhắn và gửi message tới tất cả thành viên trong nhóm	chat_group()	1
7	**Quản lý Bạn bè**	Gửi/nhận FRIEND_REQ, hiển thị danh sách bạn bè online/offline	friend_menu(), send_friend_request()	2
8	**Lưu Tin Offline**	Đồng bộ và hiển thị tin nhắn cũ ngay khi đăng nhập. **Lưu trữ lịch sử chat cục bộ.**	sync_offline_messages(), save_client_log()	1

9	**Gửi/Nhận File Nhỏ**	Triển khai cơ chế truyền các file định dạng text/image nhỏ **(Client-side)**	send_file(), recv_file()	1
10	**Đăng Xuất**	Gửi yêu cầu LOGOUT tới Server và đóng socket an toàn	logout(), close_socket()	1

Ảnh sơ đồ Protocol State Machine



lll. Kế hoạch thực hiện

Giai đoạn	Thời gian	Nội dung công việc	Kết quả mong đợi
Checkpoint 1 (hiện tại)	Tuần này	- Xây dựng kiến trúc - Xác định protocol - Chia chức năng & barem	Báo cáo + kế hoạch cụ thể

Checkpoint 2 (demo)	Tuần 11	- Cài đặt các chức năng chính: • Đăng nhập / đăng ký • Chat 1–1 • Ghi log / offline • Danh sách online - Demo thực tế	