Báo cáo Chi tiết Kỹ thuật & Đánh giá Rủi ro (Technical Detail & Impact Assessment)

Phạm vi: Các thay đổi từ commit `4759afb8` đến bản hiện tại.
Mục tiêu: Tối ưu hiệu năng (CLS, Render), xử lý lỗi UX (Input Focus, Flickering), và sửa lỗi logic Business (Tab Post, Search).

1. Chi tiết Thay đổi Kỹ thuật (Technical Details)

A. Module: Input & Focus Layout (Critical UX)

Vấn đề người dùng đang gõ tìm kiếm bị mất focus và danh sách bị giật.

File	Function / Component	Chi tiết Thay đổi (Logic/Code)
`src/service/function/index.ts`	`selectConversation(conversation, is_disable_focus)`	**Logic Flow Update:** - Thêm Guard Clause: Kiểm tra `conversationStore.option_filter_page_data.se` - Nếu đang có text search VÀ `is_disable_focus` k phải là `false` (explicit manual click) thì ép buộc `is_disable_focus = true`.
`Conversation.vue`	`pickFirstConversationImmediately()`	**Refactor & Logic:** - Gọi `selectConversation(FIRST_CONV, true)` (tr tham số `true` explicit). - Thêm logic `else` (Empty State): Nếu `size(CONVERSATIONS) == 0` thì set `select_conversation = undefined` và gọi `setParamChat` để xóa URL params.
`Conversation.vue`	`getConversation()`	**Logic Flow Update:** - Thêm điều kiện: `if (!search) is_loading_list = !!is_first_ti` - **Không** set `is_loading_list = true` khi đang ở độ search.
`ConversationItem.vue`	`clickConversation()`	**Logic Invocation:** - Gọi `selectConversation(CONVERSATION, false)`

B. Module: Layout Performance (CLS & Rendering)

Vấn đề giật khung hình (Layout Shift) khi tải trang hoặc chuyển tab.

File	Function / Component	Chi tiết Thay đổi (Template/CSS)	Mục đích
`Header.vue`	`template`, `watch`	**Reactivity & Template:** - **Removed Skeleton (v-if/else):** Thay bằng hiển thị tĩnh hoặc CSS `opacity`. - **Cache `org_name`:** Lưu tên tổ chức vào `localStorage`. Đọc từ cache ngay khi mount. - Chuyển `computed` sang `ref` + `watch(immediate: true)` cho các biến hiển thị.	**Zero CLS:** Loại bỏ hoàn toàn sự thay đổi kích thước/vị trí của Header khi tải trang. Tên tổ chức hiển thị ngay lập tức (Instant Render).
`InputChat.vue`	`template`	**Flow:** Thay thế toàn bộ `v-if/v-else` bằng `v-show`.	Tránh việc Vue phải destroy và re-create DOM nodes của `MainInput` (component nặng). Giữ component trong bộ nhớ, chỉ toggle CSS `display`.
`MessageList.vue`	`v-for`, `v-if`	**Template structure:** Ổn định cấu trúc DOM bọc ngoài.	Giảm thiểu repaint trình duyệt khi list tin nhắn thay đổi trạng thái (loading/error).

C. Module: Business Logic (Tab & Data)

Vấn đề hiển thị sai dữ liệu giữa các Tab Chat/Post.

File	Function / Component	Chi tiết Thay đổi	Mục đích
`Conversation.vue`	`loadConversationFirstTime()`	**Logic:** Uncomment `conversationStore.conversation_list = {}`.	**Fix Data Leak:** Đảm bảo xóa sạch dữ liệu của Tab cũ (Chat) trước khi tải dữ liệu Tab mới (Post), tránh hiển thị rác.
`Conversation.vue`	`countConversation()`	**Logic:** Truyền tham số `conversation_type` ('CHAT'/'POST') vào API.	Hiển thị đúng số lượng badge (số tin nhắn chưa đọc vs số comment chưa đọc) thay vì hiển thị chung chung.
`Avatar/ClientAvatar.vue`	`template`	**Logic:** Request ảnh với query string `w={size*2}&h={size*2}`.	Hỗ trợ màn hình Retina/High-DPI (ảnh sắc nét hơn).

2. Đánh giá Rủi ro & Tác động (Impact Assessment)

Trước khi Deploy, cần lưu ý các vấn đề sau:

A. Rủi ro về Focus (Thấp - Trung bình)

- **Thay đổi**: Logic `selectConversation` trong `service/function` là thay đổi global.
- **Nguy cơ**: Nếu có một flow nào đó (ngoài Search và Click) cần auto-focus hội thoại ngay sau khi tải, logic này có thể vô tình chặn nó nếu store đang lưu trạng thái `search: 'abc'`.
- **Mitigation**: Đã kiểm tra các luồng gọi chính (`ConversationItem`, `pickFirst`). Logic hiện tại ưu tiên "Safety" (không cướp focus) hơn là "Convenience" (auto focus). Đây là đánh đổi chấp nhận được để fix lỗi UX khó chịu.

B. Rủi ro về Hiệu năng Memory (Thấp)

- **Thay đổi**: `v-show` thay cho `v-if` ở `InputChat`.
- **Nguy cơ**: DOM node luôn tồn tại trong memory dù người dùng không nhìn thấy (ví dụ Input chat thường vs Input file).

- **Đánh giá**: `InputChat` chỉ có vài component, overhead bộ nhớ là không đáng kể so với lợi ích mượt mà về CPU/Rendering mà nó mang lại.

C. Rủi ro về Lifecycle (cần lưu ý khi test)

- **Thay đổi**: Các component con trong `InputChat` sẽ chạy `onMounted` ngay khi trang tải xong, thay vì chờ điều kiện `v-if`.
- **Nguy cơ**: Nếu các component này có gọi API trong `onMounted`, nó sẽ gọi ngay lập tức.
- **Kiểm tra**: Đã rà soát `MainInput`. Không có API call blockers trong `onMounted`. Logic khởi tạo là an toàn.

3. Khuyến nghị Test (QA Checklist)

Người test (QA) cần focus vào các kịch bản sau để đảm bảo an toàn:

1. **Search Input**:
   - ☐ Gõ từ khóa -> Đảm bảo con trỏ **vẫn nằm trong ô tìm kiếm**, không bị nhảy xuống ô chat.
   - ☐ Gõ từ khóa -> Danh sách hội thoại cập nhật **mượt**, không bị chớp trắng (skeleton).
   - ☐ Gõ từ khóa vô nghĩa (không có kết quả) -> Center content phải trống (hoặc ẩn), không hiện hội thoại cũ.
2. **Manual Selection**:
   - ☐ Click vào một hội thoại bất kỳ -> Ô chat phải được **focus ngay lập tức**, gõ được phím luôn.
3. **Tab Switching**:
   - ☐ Chuyển từ Chat sang Post -> Danh sách phải clear và load lại đúng định dạng bài viết.
4. **Reload Page**:
   - ☐ F5 lại trang -> Header (Tên Org) phải hiện ngay, không bị giật layout.

Kết luận Deploy:
Các thay đổi chủ yếu là **Safe Refactor** và **UX Patching**. Logic core của hệ thống (API calls, data processing) không bị thay đổi cấu trúc lớn. Rủi ro regression (lỗi lặp lại) là thấp nếu pass qua checklist trên.

D. Module: API Optimization & Initial Load Logic (New)

Vấn đề API `read_message` gọi trùng lặp (4 lần) và Thứ tự hiển thị hội thoại sai logic khi load lần đầu.

File	Function / Component	Chi tiết Thay đổi	Mục đích
`MessageList.vue`	`watch(select_conversation)`	**Optimization:** - Thêm check `new_val?.data_key === old_val?.data_key`. - Nếu key giống nhau thì `return`.	**Giảm 50% API Calls:** Ngăn chặn việc load lại tin nhắn không cần thiết khi object hội thoại thay đổi reference (do API update) nhưng bản chất vẫn là hội thoại đó. Fix lỗi `read_message` gọi 4 lần -> 2 lần.
`Conversation.vue`	`getConversation()`	**Logic Logic:** - Load lần đầu (`is_first_time`): 1. Ưu tiên thứ tự từ API trả về (`CONVERSATIONS`). 2. Nếu hội thoại đang chọn (`CURRENT_SELECTED`) **không** nằm trong list API -> Đưa xuống cuối. 3. Nếu có trong list -> Update reference và hiển thị đúng vị trí API. - Load các lần sau: Merge bình thường.	**Fix Initial Order:** - Đảm bảo hội thoại được chọn (từ URL/Search) hiển thị đúng context. - Nếu nó "cũ" (ngoài page 1) -> hiện cuối để tracking. - Nếu nó "mới" (trong page 1) -> hiện đúng thứ tự thời gian.

E. Module: UI Restoration (Testing Revert)

Đã khôi phục lại toàn bộ UI của `MessageList` theo yêu cầu (Avatars, DoubleCheck, Read Status) sau khi validation performance.

File	Component	Chi tiết
`MessageList.vue`	Template	- Khôi phục `ClientAvatar`, `PageStaffAvatar`. - Khôi phục `DoubleCheckIcon`, `ClientRead`, `StaffRead`. - Sửa lỗi typo class `mesage-client-read`.