Frontend Error Handling & UX Guide

This document outlines the standard procedures for handling errors in the ft_transcendence frontend.
Core Principle: Never leave the user guessing. Every error (network, security, or logic) must provide visible feedback.

1. Global Error Handling (The Foundation)

Context: These rules apply to every single fetch request in the application.
Files: dashboard.ts (fetchProfile), api.ts (if you use a wrapper).

HTTP Code	Root Cause	UX Action (Visual)	Technical Implementation
401 Unauthorized	JWT Token is expired, missing, or manipulated.	**Redirect to Login.**	localStorage.clear(); Maps('/login');
500+ Server Error	The Node.js backend crashed or the Database is unreachable.	**Toast Notification.**	Display a generic error toast: *"Server error. Please try again later."*
Network Error	User is offline or DNS failed.	**Toast Notification.**	Catch block: showToast("Connection failed. Check internet.", "error");

2. Authentication (Login & Sign Up)

File: pages/sendData.ts
Goal: Specific, helpful inline errors instead of generic alerts.

HTTP Code	Root Cause	UX Action	Code Logic
400 Bad Request	Validation failed (empty fields, bad email format).	**Inline Error.**	Target the <p class="error"> tag below the form. Text: *"Please check your input format."*

401 Unauthorized	User exists, but password is wrong.	**Inline Error.**	**Do NOT redirect.** Clear password field. Text: *"Invalid credentials."*
409 Conflict	User tries to register an email/username that exists.	**Inline Error.**	Text: *"Username or Email already taken."*

3. Search Bar (Security & WAF Blocks)

File: components/searchbar.ts
Context: This is the #1 target for ModSecurity. We must distinguish between "No Results" and "You are being blocked".

HTTP Code	Root Cause	UX Action	Code Logic
403 Forbidden	**WAF BLOCK.** User typed SQL (' OR 1=1) or Script (<script>) characters.	**Warning Tooltip.**	**CRITICAL:** Do NOT redirect to login. Stop the loading spinner. Show red text in dropdown: *"Invalid characters detected."*
200 OK (Empty)	Search returned valid JSON [] (No matches).	**Empty State.**	Show "No users found" inside the dropdown.
Network Error	Internet disconnected.	**Offline State.**	Show "Offline. Cannot search." in dropdown.

4. Profile Page (Navigation)

Files: dashboard.ts, profile.ts

Context: User navigates to /profile/:id.

HTTP Code	Root Cause	UX Action	Code Logic
404 Not Found	The User ID in the URL does not exist in the DB.	**404 Component.**	**Do NOT use alert().** Replace #dashboard-content with the render404Component() (see below).
500 Server Error	DB failed to fetch stats.	**Error Skeleton.**	Show a "Failed to load profile" card inside the main view.

5. Global Chat (Silent Failures)

File: chat/globalChat.ts
Context: WebSockets can fail silently. The UI must reflect connection state.

Event / Code	Root Cause	UX Action	Code Logic
Fetch History Fails	fetchPreviousMessages returns 500/Network Error.	**Inline Retry.**	Do not show empty chat. Show a button: *"Failed to load history. [Retry]"*
Socket Close/Error	Server restarted or Network lost (socket.onclose).	**Offline Indicator.**	1. Change "Live" green dot to **Red**. 2. input.disabled = true; 3. Show text: *"Reconnecting..."*
401 (WS Handshake)	Token invalid during WS connection.	**Redirect.**	If event.code === 1008 or similar policy violation,

			redirect to login.

6. Settings (Critical Actions)

File: pages/settings.ts

HTTP Code	Root Cause	UX Action	Code Logic
401 Unauthorized	User entered wrong "Current Password" during password change.	**Inline Error.**	Red text under "Current Password" input: *"Incorrect password."*
403 Forbidden	**WAF BLOCK.** User tried to save XSS/HTML in their Bio.	**Security Toast.**	Show error toast: *"Bio contains illegal characters."* **Do NOT redirect.**
200 OK	Success.	**Success Toast.**	Show green toast: *"Settings saved successfully!"*

7. Friends List (Right Panel)

File: components/rightPanel.ts

Scenario	UX Action	Code Logic
Fetch Fails	**Error Text.**	Currently, error returns []. You must distinguish between [] (0 friends) and Error. If error, show: *"Could not load friends list."*

Implementation Toolkit

1. The Toast Component

Copy this to components/toast.ts

```
export function showToast(message: string, type: 'success' | 'error' = 'error') {
   const toast = document.createElement('div');
   const colorClass = type === 'error' ? 'bg-red-600' : 'bg-green-600';

   toast.className = `fixed bottom-5 right-5 px-6 py-3 rounded-lg shadow-lg text-white
font-bold
                  transform transition-all duration-300 translate-y-10 opacity-0 z-[9999]
${colorClass}`;
   toast.innerText = message;

   document.body.appendChild(toast);

   // Animate In
   requestAnimationFrame(() => {
      toast.classList.remove('translate-y-10', 'opacity-0');
   });

   // Remove after 3s
   setTimeout(() => {
      toast.classList.add('translate-y-10', 'opacity-0');
      setTimeout(() => toast.remove(), 300);
   }, 3000);
}
```

2. The 404 Component

Add to components/errorsHandler.ts

```
export function render404Component(): string {
   return /* html */`
      <div class="flex flex-col items-center justify-center w-full h-[60vh] text-center">
         <h1 class="text-6xl font-bold text-color1 mb-4">404</h1>
         <p class="text-xl text-txtColor mb-8">The resource you are looking for does not
exist.</p>
         <button onclick="window.history.back()"
            class="px-6 py-3 bg-color1 text-black font-bold rounded-xl hover:scale-105
```

```
transition-transform">
            Go Back
          </button>
        </div>
    `;
}
```

3. Example Usage (Search Bar)

```
// Inside searchbar.ts
try {
    const response = await fetch(...)

    // WAF BLOCK HANDLING
    if (response.status === 403) {
        showTooltip("Invalid characters detected"); // Custom UI logic
        return;
    }

    // ... rest of logic
} catch (err) {
    // SILENT FAILURE HANDLING
    showDropdownError("Network Error");
}
```