



# RELATÓRIO DE VALIDAÇÃO - SPRINT 73 - FALHA CRÍTICA

**Data:** 21 de novembro de 2025  
**Validador:** Sistema de Testes Automatizado  
**Sprint:** 73 (13ª tentativa de correção do Bug #3)



## SUMÁRIO EXECUTIVO

Status Geral: **✗ FALHA CRÍTICA - BUG #3 PERSISTE APÓS 13 SPRINTS**

A validação da Sprint 73, que afirmava ter identificado a causa raiz (console.logs dentro de useMemo) e corrigido o problema, revelou que o **React Error #310 continua a ocorrer** na página de Analytics.



### Status do Problema Crítico:

Problema	Status Afirmado (Dev)	Status Real (Validação)	Resultado
Bug #3 Analytics	✓ RESOLVIDO	✗ NÃO RESOLVIDO	FALHA



## ANÁLISE DETALHADA

### Bug #3 - Analytics (React Error #310)

Status: **✗ NÃO RESOLVIDO APÓS 13 SPRINTS**





#### Evidência:

Plain Text

```
Error: Minified React error #310
at Object.Cu [as useEffect]
at i (http://localhost:3001/assets/Analytics-UhXqgaYy.js:1:7353 )
```

#### Análise:



-  Novo build está carregado ( Analytics-UhXqgaYy.js da Sprint 73)
-  **React Error #310 ainda persiste**
-  O erro ocorre no mesmo lugar (useEffect, linha 7353)
-  A página mostra "Erro ao Carregar Página"

**Conclusão:** A remoção dos console.logs **NÃO resolveu o problema**. A causa raiz identificada pelo desenvolvedor estava **INCORRETA**.



## HISTÓRICO COMPLETO DE FALHAS (SPRINTS 55-73)

Sprint	Tentativa de Solução	Resultado
55-64	Hoisting de componentes	 FALHA
65-66	useMemo para calculateStats/Health	 FALHA
67-69	Memoização de arrays extraídos	 FALHA
70	Desabilitar refetchInterval	 FALHA
71	Memoização de chart data	 FALHA
71.1	Remover dependências + extrair valores	 FALHA
72	Reverter para Sprint 67-68	 FALHA
73	<b>Remover console.logs</b>	 FALHA

**Total de Sprints Desperdiçados: 13**

**Taxa de Sucesso: 0%**



## ANÁLISE DA CAUSA RAIZ REAL

Após 13 sprints de tentativas falhadas, é evidente que:

1. **A causa raiz NUNCA foi identificada corretamente**
2. **Todas as soluções foram baseadas em suposições incorretas**
3. **O problema é mais profundo do que memoização ou console.logs**



## Possíveis Causas Reais (não exploradas):

1. **Problema no useEffect:** O erro sempre aponta para useEffect. Pode haver um loop causado por:
    - Dependências que mudam constantemente
    - setState sendo chamado dentro do useEffect sem condições adequadas
    - Cleanup function incorreta
  2. **Problema nas queries tRPC:** As queries podem estar em um estado de refetch infinito
  3. **Problema no ErrorBoundary:** O próprio ErrorBoundary pode estar causando re-renders
  4. **Problema de Estado Global:** Algum contexto React ou estado global mudando constantemente
- 



## RECOMENDAÇÕES FINAIS E DEFINITIVAS

É NECESSÁRIO PARAR DE TENTAR SOLUÇÕES ALEATÓRIAS.

### Ações Obrigatórias:

1. **USAR AMBIENTE DE DESENVOLVIMENTO NÃO MINIFICADO**
    - Configurar `NODE_ENV=development`
    - Obter mensagens de erro completas do React
    - Identificar exatamente qual linha de código está causando o loop
  2. **USAR REACT DEVTOOLS PROFILER**
    - Gravar uma sessão de profiling
    - Identificar qual componente está re-renderizando infinitamente
    - Identificar qual prop/state está mudando
  3. **ADICIONAR LOGS DETALHADOS**
    - Logar TODAS as execuções de useEffect
    - Logar TODAS as mudanças de dependências
    - Logar TODAS as execuções de useMemo
  4. **CONSIDERAR REESCRITA COMPLETA**
    - Se após os passos 1-3 a causa não for identificada em 2 horas
    - Reescrever o componente Analytics do zero
    - Usar uma arquitetura mais simples e testável
-



## DECLARAÇÃO FINAL

**Após 13 sprints e 0% de taxa de sucesso, é imperativo que a equipe de desenvolvimento:**

1. Reconheça que a abordagem atual NÃO está funcionando
2. Adote ferramentas profissionais de debug (React DevTools, ambiente dev)
3. Pare de fazer suposições sobre a causa raiz
4. Identifique a causa raiz REAL através de evidências concretas

**Status:**  **AGUARDANDO UMA MUDANÇA RADICAL NA ABORDAGEM DE DEBUGGING.**

---

**Nota:** Este é o **13º relatório de validação consecutivo** reportando o mesmo bug. A definição de insanidade é fazer a mesma coisa repetidamente e esperar resultados diferentes.