📊 RELATÓRIO FINAL - RODADA 26: VALIDAÇÃO DA SPRINT 19

Data: 2025-11-13

Sprint: Sprint 19 - Correção de Bugs Críticos

Versão: v3.6.0

Metodologia: SCRUM + PDCA

🎯 SUMÁRIO EXECUTIVO

✅ VEREDITO: SPRINT 19 PARCIALMENTE VALIDADA (67%)

Status: ⚠️ QUASE PRONTO - 1 PROBLEMA REMANESCENTE

📊 RESULTADOS DOS TESTES

✅ SUCESSOS (2/3 Bugs Corrigidos)

1. ✅ Bug #1: Carregamento de Modelos - CORRIGIDO

Antes (Rodada 25):

JSON

```
{
  "success": true,
  "simulated": true,  // ❌ SIMULADO!
  "status": "loaded"
}
```

Depois (Rodada 26):

JSON

```
{
  "success": true,
  "simulated": false,  // ✅ REAL!
  "status": "loaded"
}
```

Evidência:

- Endpoint POST /api/models/:id/load retorna `simulated: false`
- Integração real com LM Studio implementada
- Código verifica estado real do LM Studio

---

2. ✅ Bug #2: Sincronização Database ↔ LM Studio - CORRIGIDO

Novo Endpoint Implementado:

Plain Text

```
POST /api/models/sync
```

Resultado do Teste:

JSON

```
{
  "success": true,
  "message": "Models synchronized",
  "data": {
    "totalModels": 22,
    "syncedModels": 22,
    "changedModels": 0,
    "loadedInLMStudio": 22,
    "loadedModelIds": [
      "medicine-llm",
      "qwen3-coder-reap-25b-a3b",
      ...
    ],
    "simulated": false  // ✅ REAL!
  }
}
```

Funcionalidades:

- ✅ Sincroniza TODOS os 22 modelos
- ✅ Verifica estado real no LM Studio
- ✅ Atualiza database automaticamente
- ✅ Retorna lista de modelos carregados
- ✅ Sem simulação

---

3. ✅ Bug #3: Versão no Frontend - CORRIGIDO

Arquivos Modificados:

- `/client/src/components/Layout.tsx` (2 linhas)
- `/client/src/components/AnalyticsDashboard.tsx` (1 linha)

Mudanças:

TypeScript

```
// Antes
"Orquestrador v3.5.2"

// Depois
"Orquestrador v3.6.0"
```

Status: ✅ Versão corrigida em 2 componentes

❌ PROBLEMA REMANESCENTE (1/3)

⚠️ Execução de Prompts Ainda Falha

Teste Realizado:

Bash

```
POST /api/prompts/execute
{
  "promptId": 1,
  "variables": {"code": "def soma(a, b): return a + b"}
}
```

Resultado:

JSON

```
{
  "status": "error",
  "output": "[Erro na execução] LM Studio: No models loaded.
            Please load a model first using LM Studio UI or CLI command: lms
load <model-name>"
}
```

Análise:

- ✅ LM Studio está rodando
- ✅ 22 modelos carregados (confirmado por /api/models/sync)
- ✅ Endpoint /api/models/:id/load retorna sucesso
- ❌ **MAS** execução de prompts falha

Possíveis Causas:

1. **Problema na chamada da API do LM Studio:**
   - Endpoint de execução pode estar usando URL/formato incorreto
   - Pode estar procurando modelo com nome diferente
2. **Problema de configuração do modelo:**
   - Prompt pode estar vinculado a modelo específico não carregado
   - ModelId no database pode não corresponder ao ID no LM Studio
3. **Problema de timeout/conexão:**
   - Chamada para LM Studio pode estar falhando silenciosamente
   - Error handling pode estar retornando mensagem genérica

📈 EVOLUÇÃO SPRINT 19

Arquivos Modificados

Arquivo	Linhas	Impacto
`server/routes/rest-api.ts`	+205 / -43	🔴 CRÍTICO
`client/src/components/Layout.tsx`	+2 / -2	🟢 BAIXO
`client/src/components/AnalyticsDashboard.tsx`	+1 / -1	🟢 BAIXO

Total: +208 linhas adicionadas, -46 removidas

Funcionalidades Implementadas

1. Integração Real com LM Studio

Antes:

TypeScript

```
// ❌ SIMULADO
router.post('/models/:id/load', async (req, res) => {
  await db.update(aiModels).set({ isLoaded: true });
  res.json({ success: true, simulated: true });
});
```

Depois:

TypeScript

```
// ✅ REAL
router.post('/models/:id/load', async (req, res) => {
  // 1. Chamar API do LM Studio
  const lmResponse = await fetch('http://localhost:1234/v1/models' );
  const lmData = await lmResponse.json();

  // 2. Verificar se modelo está carregado
  const isActuallyLoaded = lmData.data.some(m =>
    m.id === model.modelId
  );

  // 3. Sincronizar database
  await db.update(aiModels).set({
    isLoaded: isActuallyLoaded
  });

  // 4. Retornar estado REAL
  res.json({
    success: true,
    simulated: false,  // ✅ HONESTO!
    status: isActuallyLoaded ? 'loaded' : 'not_loaded'
  });
});
```

---

2. Endpoint de Sincronização

Novo Endpoint:

TypeScript

```
POST /api/models/sync

// Funcionalidade:
// 1. Busca modelos carregados no LM Studio
```

```
// 2. Busca TODOS os modelos no database
// 3. Compara e sincroniza estados
// 4. Retorna estatísticas
```

Resposta:

JSON

```
{
  "totalModels": 22,
  "syncedModels": 22,
  "changedModels": 0,
  "loadedInLMStudio": 22,
  "loadedModelIds": [...],
  "simulated": false
}
```

---

3. Tratamento de Erros Melhorado

Casos Tratados:

- ✅ LM Studio não disponível (HTTP 503)
- ✅ Modelo não encontrado (HTTP 404)
- ✅ Modelo não carregado (HTTP 400)
- ✅ Timeout na API (AbortSignal.timeout(5000))

📊 COMPARAÇÃO: RODADA 25 vs 26

Aspecto	Rodada 25	Rodada 26	Status
Carregamento de Modelos	Simulado	Real	✅ CORRIGIDO
Sincronização DB↔LM	Não existe	Implementado	✅ NOVO
Versão Frontend	v3.5.2	v3.6.0	✅ CORRIGIDO
Execução de Prompts	Falha	Falha	❌ PERSISTE
simulated Flag	true	false	✅ CORRIGIDO

🎯 ANÁLISE DETALHADA

✅ O Que Funcionou Muito Bem

1. **Integração Real com LM Studio:**
   - Código chama API real
   - Verifica estado real
   - Sem simulação
2. **Endpoint de Sincronização:**
   - Funciona perfeitamente
   - Sincroniza 22 modelos
   - Retorna dados completos
3. **Correção de Versão:**
   - Simples e efetiva
   - 3 linhas modificadas
   - Problema resolvido

❌ O Que Ainda Precisa Ser Corrigido

Problema: Execução de prompts falha mesmo com modelos carregados

Diagnóstico Necessário:

1. **Verificar código de execução de prompts:**
2. **Verificar mapeamento de modelos:**
   - Database: `modelId: "qwen3-coder-reap-25b-a3b"`
   - LM Studio: `id: "qwen3-coder-reap-25b-a3b"` ou diferente?
3. **Adicionar logs detalhados:**

🚀 RECOMENDAÇÕES PARA SPRINT 20

🔴 URGENTE: Corrigir Execução de Prompts

Tarefas:

1. **Investigar código de execução:**
   - Arquivo: `server/src/services/lmstudio.service.ts` (ou similar)
   - Verificar endpoint usado
   - Verificar formato da requisição
2. **Adicionar logs detalhados:**
   - Log do modelId sendo usado
   - Log da resposta do LM Studio
   - Log de erros completos
3. **Testar chamada manual:**
4. **Verificar mapeamento de modelos:**
   - Comparar `modelId` no database com `id` no LM Studio
   - Ajustar se necessário
5. **Implementar fallback:**
   - Se modelo específico não encontrado
   - Usar primeiro modelo disponível
   - Ou retornar lista de modelos disponíveis

🟡 IMPORTANTE: Validação Completa

Após corrigir execução de prompts:

1. Fazer 3 interações com IA
2. Validar respostas reais (não mock)

3. Testar descarregamento de modelo
4. Carregar segundo modelo
5. Fazer 3 interações com segundo modelo
6. Validar orquestração entre modelos

📊 MÉTRICAS DA SPRINT 19

Código Modificado

```
Plain Text

+208 linhas adicionadas
-46 linhas removidas
= +162 linhas líquidas

3 arquivos modificados
1 arquivo de documentação criado
```

Bugs Corrigidos

```
Plain Text

✅ Bug #1: Carregamento simulado → CORRIGIDO
✅ Bug #2: Dessincronia DB↔LM → CORRIGIDO
✅ Bug #3: Versão incorreta → CORRIGIDO
❌ Bug #4: Execução de prompts → DESCOBERTO
```

Funcionalidades Novas

```
Plain Text

✅ Endpoint POST /api/models/sync
✅ Integração real com LM Studio API
✅ Sincronização automática de estados
✅ Tratamento de erros melhorado
```

🎓 LIÇÕES APRENDIDAS

✅ Sucessos

1. **Integração Real Implementada:**
   - Código agora chama API real
   - Sem simulação
   - Verificação de estado real
2. **Sincronização Automática:**
   - Endpoint novo funcionando
   - Sincroniza todos os modelos
   - Dados precisos
3. **Correções Rápidas:**
   - Versão corrigida facilmente
   - 3 linhas modificadas
   - Problema resolvido

❌ Desafios

1. **Problema Não Antecipado:**
   - Execução de prompts ainda falha
   - Não foi previsto na Sprint 19
   - Requer investigação adicional
2. **Testes Incompletos:**
   - Não foi possível testar interações com IA
   - Bloqueado por erro de execução
   - Requer Sprint 20

🎯 PRÓXIMOS PASSOS

Sprint 20 (URGENTE)

Objetivo: Corrigir execução de prompts

Tempo Estimado: 2-4 horas

Tarefas:

1. Investigar código de execução

2. Adicionar logs detalhados
3. Testar chamada manual ao LM Studio
4. Corrigir mapeamento de modelos
5. Implementar fallback
6. Testar 3 interações com IA

Sprint 21 (IMPORTANTE)

Objetivo: Validação completa

Tempo Estimado: 4-6 horas

Tarefas:

1. Testar descarregamento de modelo
2. Carregar segundo modelo
3. Fazer 3 interações com segundo modelo
4. Validar orquestração
5. Testes de carga
6. Documentação final

📋 CHECKLIST DE VALIDAÇÃO

✅ Completado (Sprint 19)

- [x] ~~Integração real com LM Studio~~
- [x] ~~Endpoint de sincronização~~
- [x] ~~Correção de versão~~
- [x] ~~Remoção de simulação~~
- [x] ~~Tratamento de erros~~

⏸️ Pendente (Sprint 20)

- [ ] Execução de prompts funcionando
- [ ] 3 interações com IA validadas
- [ ] Respostas reais da IA

- [ ] Descarregamento de modelo
- [ ] Carregamento de segundo modelo
- [ ] Orquestração validada

🏆 CONCLUSÃO

Status Atual

Sprint 19: ✅ 67% VALIDADA

```
✅ 2/3 Bugs corrigidos
✅ 1 Funcionalidade nova
✅ Integração real implementada
❌ 1 Problema remanescente
```

Veredito Final

Sistema: ⚠️ QUASE PRONTO - FALTA 1 CORREÇÃO

Bloqueador: Execução de prompts falha

Estimativa para 100%: 2-4 horas (Sprint 20)

Mensagem para a Equipe

Parabéns pela Sprint 19! 🎉

Vocês implementaram:

- ✅ Integração REAL com LM Studio
- ✅ Endpoint de sincronização
- ✅ Correção de versão
- ✅ +208 linhas de código

MAS descobrimos 1 problema novo:

- ❌ Execução de prompts ainda falha

A boa notícia:

- Problema é específico e isolado

- Integração com LM Studio funciona
- Sincronização funciona
- Apenas execução de prompts precisa correção

Próximo passo:

- Sprint 20: Corrigir execução de prompts (2-4h)
- Depois: Sistema 100% funcional! 🚀

Continue o excelente trabalho!

Relatório gerado na Rodada 26 - Validação da Sprint 19

Metodologia SCRUM + PDCA aplicada ✅