

WEB STAR STUDIO

MANUAL

MÉTODO DE DESENVOLVIMENTO DE SITES

MANUAL 6

Testes, QA, Debugging, Segurança e SEO

VERSÃO

v1.1 FINAL

DATA

11/09/2026

STATUS

FINAL — APROVADO

MANUAL

MANUAL 6

Testes, QA, Debugging, Segurança e SEO

VERSÃO	v1.1 FINAL
DATA	11/09/2026
STATUS	FINAL — APROVADO
ESCOPO DOCUMENTAL	primeira documentação ampliada — P0 + P1 + P2 relevante
PROJETO	WSS — Método de Desenvolvimento de Sites
FORMALIZAÇÃO	Pacote Corretivo Pós-Entrega v1.1 aprovado pela Coordenação em 11/09/2026; versão canônica vigente, com v1.0 FINAL preservado como baseline histórico.

NAVEGAÇÃO DO DOCUMENTO

Sumário

Estrutura do documento canônico v1.1 FINAL.

1. Objetivo	8
2. Escopo e fronteira	9
3. Posição deste Manual no fluxo	10
PARTE I — ESTRATÉGIA DE TESTES AS-IS	11
4. Modelo atual	12
5. QA não significa apenas teste automatizado	12
6. Codex em testes	12
7. Build e typecheck	13
8. Testes automatizados	13
PARTE II — MOMENTO PRINCIPAL DE QA	14
9. Quando ocorre	15
10. Relação com revisão do cliente	15
PARTE III — QA VISUAL E RESPONSIVO	16
11. Três cenários principais	17
12. Aprovação visual prática	17
13. O que o Dev observa	17
14. Responsividade	18
15. Viewports intermediários	18
PARTE IV — QA FUNCIONAL DO SITE PÚBLICO	19
16. Percurso página por página	20
17. Itens principais	20
18. Links e rotas	20
19. Formulários e Leads	20
20. Chat	21
21. Chatbot	21
PARTE V — QA DO ADMIN	22

22. Módulo por módulo	23
23. Itens típicos	23
24. Perfis	23
PARTE VI — ESTADOS DE ERRO E CASOS NÃO SISTEMÁTICOS	24
25. Estados vazios	25
26. Mensagens de erro	25
27. 404	25
PARTE VII — DADOS DE TESTE	26
28. Uso de dados falsos	27
29. Limpeza	27
PARTE VIII — DEBUGGING	28
30. Sequência prática	29
31. Começar pelo sintoma	29
32. Console e DevTools	29
33. Frontend/visual	29
34. Backend, banco e autenticação	30
35. Problemas difíceis	30
36. Causa confirmada no AS-IS	31
PARTE IX — CASO REAL DE DEBUGGING: MIRELA	32
37. Sintoma	33
38. Evidência	33
39. Diagnóstico	33
40. Correção	33
41. Lição	33
PARTE X — REGRESSÃO	35
42. Não existe checklist universal	36
43. Alteração visual	36
44. Alteração funcional	36
45. Admin	36
46. Autenticação	36
47. Base/portabilidade	37
48. Regressão não óbvia	37

PARTE XI — TROUBLESHOOTING COMO MÉTODO	38
49. Padrão transversal observado	39
50. Troubleshooting ≠ catálogo neste Manual	39
51. Casos que alimentam o método	39
PARTE XII — SMOKE BUILD DE PRODUÇÃO	41
52. Caso Guilherme Mendonça	42
53. Resposta técnica observada	42
54. Rotas observadas no caso	42
55. O que o caso prova	42
56. Limite documental	43
PARTE XIII — SEO TÉCNICO AS-IS	44
57. Como o Dev trabalha SEO hoje	45
58. Auditoria não é ordem automática	45
59. Classificação de achados	45
60. Itens SEO observados	45
61. Ajustar somente o necessário	46
62. Validação pós-ajuste SEO	46
PARTE XIV — SEOMATOR, AEO E GEO	47
63. SEOmator	48
64. Regra de uso	48
65. Falso positivo ou limitação de coleta	48
66. Dimensões AEO/GEO observadas externamente	48
67. Checklist AEO/GEO real em projeto	49
68. Limite do checklist Daniel	50
PARTE XV — NORMA TÉCNICA VIGENTE DE SEO/AEO/GEO	51
69. Camada normativa separada	52
70. Regra obrigatória de separação	53
PARTE XVI — SEGURANÇA AS-IS	54
71. Como segurança aparece hoje	55
72. Hardening real da Base Admin	55
73. Papéis	55
74. UI não basta	56

75. RLS	56
76. PII	56
77. Superfície pública legítima	56
78. Secrets	57
79. Exemplo real de hardening	57
80. Padrão de segurança observado	58
PARTE XVII — VALIDAÇÃO DE SEGURANÇA PROPORCIONAL	59
81. Procedimento reproduzível sustentado	60
82. Por que não vira SOP próprio agora	60
PARTE XVIII — FEEDBACK E RODADAS DE CORREÇÃO	61
83. Canal predominante	62
84. Formatos	62
85. Consolidação	62
86. Quando voltar para gestão	62
87. ajustes.md	63
88. Antes/depois	63
89. Relação com regressão	63
PARTE XIX — AUSÊNCIAS ATUAIS DE PADRONIZAÇÃO	64
90. Lista consolidada deste Manual	65
PARTE XX — OPORTUNIDADES FUTURAS — FORA DO ESCOPO	66
91. Oportunidades identificáveis	67
PARTE XXI — RELAÇÃO COM OS OUTROS MANUAIS	68
92. Relação com o Manual 3	69
93. Relação com o Manual 4	69
94. Relação com o Manual 5	69
95. Relação com o Manual 7	69
96. Relação com o Manual 8	70
PARTE XXII — SOPS DERIVADOS	71
97. SOP-008 — Executar QA Visual, Funcional e Responsivo	72
98. SOP-009 — Diagnosticar e Corrigir Bug por Camada	72
99. SOP-010 — Tratar Auditoria SEO, AEO e GEO	72
100. SOP-011 — Processar Rodada de Feedback via ajustes.md	73

101. Procedimentos incorporados, sem SOP próprio	73
PARTE XXIII — CASOS REAIS UTILIZADOS	74
102. Guilherme Mendonça	75
103. Daniel P	75
104. Mirela Albuquerque	75
105. Guilherme Maia	75
PARTE XXIV — CRITÉRIO DE CONCLUSÃO	76
106. Critério de conclusão deste Manual	77
107. Fontes principais desta versão	77

1. Objetivo

Este Manual documenta como a WSS verifica, investiga e corrige a qualidade técnica e funcional dos sites durante o desenvolvimento.

Seu foco é a passagem:

```
implementação
→ QA
→ teste funcional
→ detecção de problema
→ debugging
→ correção
→ regressão
→ auditorias técnicas
→ ajustes
→ nova validação
```

Ao final deste Manual, o leitor deve compreender:

- quando ocorre o principal QA;
- como o Dev combina validação manual e Codex;
- quais dispositivos/tamanhos são priorizados;
- como o site público é percorrido;
- como o Admin é testado;
- como Leads, formulários, Chat, Chatbot e perfis entram no QA;
- o que não é testado sistematicamente hoje;
- como o Dev inicia um debugging;
- quando usa console/DevTools, Codex, Claude ou Lovable;
- como considera uma hipótese suficientemente confirmada;
- como revalida regressões;
- por que “dev funciona” não prova saúde do build de produção;
- como auditorias SEO/AEO/GEO são interpretadas;
- como SEOmator é usado como input e não como autoridade automática;
- como segurança aparece nas auditorias reais;
- como RLS, papéis, secrets, PII e autorização server-side aparecem no AS-IS;
- como feedback é convertido em `ajustes.md`;
- quais gates formais não existem hoje e não devem ser inventados.

Este Manual registra o **AS-IS** e mantém normas técnicas vigentes em camada própria.

2. Escopo e fronteira

2.1 O que entra neste Manual

Este Manual cobre:

1. momento principal de QA;
2. teste manual;
3. Codex em testes amplos;
4. QA visual;
5. QA responsivo;
6. QA funcional;
7. site público;
8. Admin;
9. Leads/formulários;
10. Chat;
11. Chatbot;
12. autenticação/perfis;
13. dados de teste;
14. debugging;
15. console/DevTools;
16. decisão de ferramenta por camada;
17. regressão;
18. troubleshooting como método;
19. smoke build como procedimento condicional;
20. SEO técnico AS-IS;
21. AEO/GEO AS-IS;
22. SEOmator;
23. segurança AS-IS;
24. RLS;
25. papéis;
26. secrets;
27. PII;
28. autorização server-side;
29. feedback;
30. `ajustes.md`;
31. revalidação das correções.

2.2 O que não entra neste Manual

Ficam fora:

- construção visual detalhada;
- operação Git;

- configuração funcional da Base Admin;
- domínio/DNS;
- publicação em produção;
- contingência de produção;
- catálogo completo de troubleshooting;
- estudos de caso completos.

O catálogo canônico de problemas conhecidos pertence ao Manual 8.

A publicação e o QA em ambiente final pertencem ao Manual 7.

3. Posição deste Manual no fluxo

O principal recorte é:

```
páginas suficientemente desenvolvidas
→ QA/revisão do Dev + revisão interna adicional podem se sobrepor
→ correções
→ revisão do cliente quando aplicável
→ novas correções
→ auditorias técnicas aplicáveis
→ validação final
```

No AS-IS, a validação principal começa depois de as páginas estarem suficientemente desenvolvidas. Quando o Dev sinaliza uma versão ou área revisável, outras pessoas podem iniciar a revisão interna em paralelo à própria revisão do Dev.

Isso não significa que testes só aconteçam nesse momento.

Durante a construção, o Dev já valida:

- visual;
- mobile;
- ações principais;
- módulos específicos.

O M6 documenta a camada **sistemática/principal de validação e investigação**.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE I — ESTRATÉGIA DE TESTES AS-IS

4. Modelo atual

A estratégia atual combina:

- validação manual do Dev;
- pedidos amplos de teste ao Codex;
- testes técnicos específicos quando necessários;
- build/typecheck em algumas tarefas/projetos;
- inspeção de navegador/console;
- testes funcionais de fluxos completos.

Não existe uma suíte formal obrigatória e idêntica em todos os projetos.

5. QA não significa apenas teste automatizado

No método atual, QA inclui:

- olhar;
- navegar;
- clicar;
- preencher;
- comparar;
- observar;
- reproduzir;
- testar fluxos reais.

A validação humana é central.

6. Codex em testes

O Dev pode pedir ao Codex uma rodada ampla de testes.

O Codex pode contribuir com:

- navegação;
- inspeção;
- testes técnicos;
- build;
- typecheck;
- leitura de erros;
- comparação de comportamento.

Limite

A existência do Codex não elimina a validação manual do Dev.

7. Build e typecheck

Build e typecheck aparecem em trabalhos reais.

Mas:

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não formam uma suíte formal obrigatória universal que seja executada de forma idêntica em todo projeto e toda rodada.

Este Manual não inventa uma pipeline obrigatória.

8. Testes automatizados

Projetos podem conter testes ou scripts específicos.

Exemplos reais incluem:

- benchmark do Chatbot;
- smoke build de produção;
- scripts/verificações dirigidas.

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não existe suíte automatizada universal obrigatória para todos os sites WSS.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE II — MOMENTO PRINCIPAL DE QA

9. Quando ocorre

O principal QA ocorre depois de as páginas estarem suficientemente desenvolvidas. A revisão interna adicional não precisa esperar o Dev concluir toda a sua própria revisão; ela pode começar em paralelo quando houver versão revisável.

O Dev combina:

1. pedido de testes ao Codex;
2. validação manual;
3. correções;
4. revalidação.

10. Relação com revisão do cliente

O institucional pode ser publicado temporariamente para revisão do cliente enquanto o Admin continua sendo finalizado.

Portanto:

- QA do institucional;
- configuração do Admin;
- revisão do cliente;

podem ter sobreposição temporal.

O fluxo não é completamente linear.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE III — QA VISUAL E RESPONSIVO

11. Três cenários principais

O Dev prioriza três cenários:

Desktop

MacBook conectado a monitor.

Notebook

O próprio MacBook.

Mobile

Aparelho físico acessando o projeto local pela rede/Wi-Fi.

12. Aprovação visual prática

O visual é considerado adequado quando:

- desktop;
- notebook;
- mobile;

estão corretos no julgamento do Dev.

13. O que o Dev observa

O QA visual verifica, entre outros pontos:

- fidelidade;
- tipografia;
- espaçamento;
- alinhamento;
- caixas/cards;
- imagens;
- crop;
- overflow;
- quebras;
- animações;
- scroll;
- navegação;
- consistência;

- CTAs.
-

14. Responsividade

A validação responsiva observa:

- layout;
- textos;
- imagens;
- botões;
- navegação;
- efeitos;
- scroll horizontal;
- comportamento em telas menores.

O Dev pode aceitar solução diferente entre desktop e mobile.

15. Viewports intermediários

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não existe teste sistemático de uma grande matriz de viewports intermediários.

A limitação de tempo faz o Dev priorizar:

- desktop;
- notebook;
- mobile físico.

Isso não significa que tamanhos intermediários nunca sejam testados.

Significa que não há matriz formal universal.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE IV — QA FUNCIONAL DO SITE PÚBLICO

16. Percurso página por página

O site público é percorrido:

- página por página;
 - seção por seção;
 - do Hero ao Footer.
-

17. Itens principais

O Dev verifica:

- navegação;
 - links;
 - CTAs;
 - formulários;
 - interações;
 - botões;
 - fluxos de contato;
 - ações principais.
-

18. Links e rotas

Links precisam levar ao destino esperado.

Caso real Daniel P:

- CTA apontava para `/sobre-mim`;
- a rota existente era `/sobre`;
- resultado era 404.

A correção foi localizada no path.

Lição

Sofisticação visual não substitui QA básico de navegação.

19. Formulários e Leads

O teste ponta a ponta inclui:

1. acionar formulário/CTA/agendamento;

2. preencher dados;
3. conferir sucesso/erro;
4. abrir Admin;
5. localizar Lead;
6. conferir nome;
7. conferir origem/trajeto;
8. conferir especialidade/opção.

Isso valida frontend e chegada do dado ao Admin.

20. Chat

O Dev:

1. inicia conversa;
 2. verifica comportamento;
 3. abre Admin;
 4. confirma que a conversa aparece.
-

21. Chatbot

Quando aplicável, o Dev testa como:

- usuário;
- administrador.

Pode verificar:

- captura de nome/telefone;
- criação do Lead;
- pergunta coberta;
- pergunta sem resposta confirmada;
- handoff humano;
- WhatsApp quando configurado;
- aparição da conversa no Admin.

A Base possui benchmark técnico, mas ele não substitui o teste manual.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE V — QA DO ADMIN

22. Módulo por módulo

O Dev testa o Admin:

1. abre o módulo;
 2. clica;
 3. executa os fluxos relevantes;
 4. observa o resultado.
-

23. Itens típicos

Conforme o projeto:

- Blog;
 - Leads;
 - Chat;
 - Chatbot;
 - FAQ;
 - depoimentos;
 - perfis;
 - configurações;
 - outras capacidades contratadas.
-

24. Perfis

O Dev cria duas contas internas:

- Equipe/cliente;
- Webstar.

Ele verifica se cada perfil acessa somente o que deveria.

A validação técnica de autorização server-side aparece mais adiante neste Manual.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE VI — ESTADOS DE ERRO E CASOS NÃO SISTEMÁTICOS

25. Estados vazios

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Estados vazios não são testados sistematicamente em todos os projetos.

Entram quando:

- o fluxo exige;
- há risco percebido;
- surge problema;
- a auditoria aponta.

26. Mensagens de erro

Também não existe uma matriz universal de mensagens de erro testadas em toda entrega.

O Dev testa principalmente os fluxos essenciais.

27. 404

URLs inexistentes/404 não são testadas sistematicamente no AS-IS de todos os projetos.

Quando surgem em QA/auditoria, são tratadas.

Importante

Se existir **NORMA TÉCNICA VIGENTE** exigindo 404 real, isso pertence à camada normativa e deve ser respeitado quando aplicável.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE VII — DADOS DE TESTE

28. Uso de dados falsos

O Dev cria dados de teste:

- Leads;
- conversas;
- registros necessários.

Isso permite validar fluxos ponta a ponta.

29. Limpeza

Antes da entrega do Admin, os dados de teste devem ser limpos/resetados.

A execução final da limpeza pertence ao Manual 7/SOP-012.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE VII — DEBUGGING

30. Sequência prática

O debugging atual é pragmático e não rígido.

Sequência típica:

1. reproduzir no navegador;
 2. observar o sintoma;
 3. identificar a camada provável;
 4. usar ferramenta adequada;
 5. aplicar correção ligada à hipótese;
 6. reproduzir novamente.
-

31. Começar pelo sintoma

O Dev não começa necessariamente lendo código linha por linha.

Primeiro procura:

- reproduzir;
 - ver o erro;
 - entender onde quebra.
-

32. Console e DevTools

Quando o sintoma sugere:

- configuração;
- key;
- ambiente;
- erro frontend;
- runtime;

o Dev abre console/DevTools.

33. Frontend/visual

Problemas de:

- visual;
- layout;

- comportamento;
- efeito;
- responsividade;

tendem a ser tratados com:

- Codex;
 - Claude.
-

34. Backend, banco e autenticação

Esta seção trata de **investigação/correção pós-criação**. A criação inicial do projeto e da estrutura herdada de backend/banco/auth continua ocorrendo no Lovable durante o remix da Base Admin.

Em problemas pós-criação ligados a:

- backend;
- banco;
- autenticação;
- configuração integrada;

o Codex pode ser a primeira camada de investigação/condução e, conforme a necessidade, executar diretamente o que couber ou operar/orquestrar:

- Lovable;
- Lovable Cloud.

O uso direto do Lovable pelo Dev continua válido quando a integração da plataforma torna esse caminho mais adequado.

35. Problemas difíceis

Quando um problema é mais complexo, o Dev pode usar:

- Lovable;
- Codex;

em paralelo para comparar investigações.

36. Causa confirmada no AS-IS

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não existe metodologia formal universal de root cause analysis.

O Dev considera a causa suficientemente confirmada quando:

- a hipótese levou a uma correção;
- o comportamento foi restaurado;
- o erro deixou de reproduzir.

Essa é uma confirmação pragmática.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE IX — CASO REAL DE DEBUGGING: MIRELA

37. Sintoma

No projeto Mirela Albuquerque:

- ao retornar ao Lovable, a configuração/key do Supabase estava ausente/incorreta;
 - o Blog deixou de funcionar corretamente.
-

38. Evidência

O erro apareceu no console.

A implementação atual corrobora que ausência de:

- URL;
- publishable key;

pode impedir fluxos dependentes do Supabase.

39. Diagnóstico

A causa estava na camada de:

- configuração;
- ambiente;
- conexão Supabase/Lovable.

Não no componente visual do Blog.

40. Correção

A configuração foi corrigida via Lovable/ambiente.

41. Lição

Quando um módulo inteiro quebra após mudança de ambiente/conexão, verificar env/secrets/conexões antes de reescrever código funcional.

O caso completo pertence ao Manual 8.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE X — REGRESSÃO

42. Não existe checklist universal

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não há padrão único e completo de regressão para todas as mudanças.

A regressão é contextual.

43. Alteração visual

Depois de mudança visual:

- revalidar desktop;
 - revalidar mobile;
 - verificar a área relacionada.
-

44. Alteração funcional

Depois de mudança funcional:

- refazer o fluxo afetado;
 - testar entradas/saídas;
 - verificar área relacionada.
-

45. Admin

Depois de mudança no Admin:

- testar o módulo alterado;
 - verificar fluxos dependentes quando óbvios.
-

46. Autenticação

Depois de alteração de acesso/perfil:

- testar perfis;
 - confirmar limites de acesso.
-

47. Base/portabilidade

Depois de adaptar capacidade da Base:

- testar capacidade portada;
 - testar dependências relevantes;
 - preservar particularidades locais.
-

48. Regressão não óbvia

Regressões não óbvias podem aparecer somente em:

- QA posterior;
- revisão interna;
- revisão do cliente;
- produção.

Esse risco faz parte do AS-IS atual.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XI — TROUBLESHOOTING COMO MÉTODO

49. Padrão transversal observado

A VAL-004 permitiu sintetizar:

```
sintoma
→ reproduzir
→ localizar camada
→ formular hipótese
→ confirmar proporcionalmente
→ corrigir de forma localizada
→ revalidar
```

50. Troubleshooting ≠ catálogo neste Manual

Este Manual ensina **como investigar**.

O Manual 8 mantém o catálogo canônico de casos:

- TS-001;
- TS-002;
- etc.

Isso evita duplicação.

51. Casos que alimentam o método

Dev funciona / produção 500

Possível diferença entre dev e artefato real.

Auditoria externa contradiz UI

Investigar representação/crawler.

Blog quebra após ambiente

Investigar env/secrets/conexão.

CTA 404

Comparar destino com rotas.

Base ameaça legado

Comparar equivalentes/deltas/dependências.

Relatório antigo pede correção já resolvida

Confrontar achado com estado atual.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XII — SMOKE BUILD DE PRODUÇÃO

52. Caso Guilherme Mendonça

O projeto registrou incidente em que:

- modo dev funcionava;
- build de produção apresentava falha;
- rotas respondiam HTTP 500.

A causa estava ligada ao pacote de produção/bundling.

53. Resposta técnica observada

Foi criado smoke test que:

1. gera o build real;
 2. sobe localmente o artefato/worker;
 3. testa rotas públicas;
 4. aceita redirect esperado no Admin;
 5. falha em status inesperado;
 6. orienta não publicar quando falha.
-

54. Rotas observadas no caso

O smoke atual cobre:

- `/`;
 - `/blog`;
 - `/contato`;
 - `/sobre`;
 - `/especialidades`;
 - `/auth`;
 - `/admin`.
-

55. O que o caso prova

Dev saudável não garante artefato de produção saudável.

Mudanças em:

- Vite;
- Nitro;
- TanStack Start;
- bundler;
- dependências de build;

podem justificar smoke do pacote real.

56. Limite documental

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

O smoke build acima não foi comprovado como gate universal histórico de todos os sites WSS.

Portanto:

- é procedimento técnico real e valioso;
- entra neste Manual;
- não vira SOP independente nesta primeira versão;
- não deve ser apresentado como obrigação AS-IS universal.

Se norma futura o tornar obrigatório, isso será outra camada.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XIII — SEO TÉCNICO AS-IS

57. Como o Dev trabalha SEO hoje

O Dev não executa manualmente uma lista técnica completa item por item em todo projeto.

Ele tende a pedir auditoria ao:

- Lovable;
- Codex;

e trabalhar os achados.

58. Auditoria não é ordem automática

Princípio observado:

```
auditoria/relatório  
→ confrontar com estado atual  
→ classificar achado  
→ corrigir somente delta real  
→ validar
```

59. Classificação de achados

No caso Guilherme, os achados foram tratados como:

- presente;
- parcialmente resolvido;
- já resolvido;
- não reproduzido.

Essa classificação evita refazer algo já corrigido.

60. Itens SEO observados

Auditorias reais verificaram:

- title;
- meta description;
- canonical;
- Open Graph;
- Twitter Cards;

- imagem social;
 - dados estruturados;
 - breadcrumbs;
 - sitemap;
 - `robots.txt` ;
 - links internos;
 - `noindex/nofollow` ;
 - domínio oficial versus preview;
 - HTML SSR.
-

61. Ajustar somente o necessário

Exemplo real:

- canonical já resolvido → preservado;
 - Open Graph parcial → completado;
 - sitemap sem problema descrito → preservado;
 - `robots.txt` incompleto → corrigido;
 - schemas parciais → completados.
-

62. Validação pós-ajuste SEO

Foram observadas validações como:

- lint direcionado;
- diff check;
- build;
- HTML SSR;
- navegação;
- links internos;
- canonical;
- Open Graph;
- breadcrumbs;
- sitemap;
- robots.

Isso demonstra que auditoria não termina na edição.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XIV — SEOmator, AEO E GEO

63. SEOmator

SEOmator é ferramenta externa confirmada no AS-IS.

No Daniel P há auditoria GEO real.

64. Regra de uso

Resultado do SEOmator deve ser tratado como:

1. achado;
2. interpretação;
3. plano sugerido;
4. implementação comprovada ou não.

Não pular direto de "score" para "alteração".

65. Falso positivo ou limitação de coleta

Caso Daniel:

- SEOmator reportou H1/imagens/schema ausentes;
- a UI mostrava os elementos;
- o crawler não via corretamente conteúdo criado client-side.

Lição

Antes de corrigir:

- entender o que o crawler lê;
 - inspecionar HTML bruto;
 - distinguir ausência real de invisibilidade ao crawler.
-

66. Dimensões AEO/GEO observadas externamente

A auditoria considerou:

- AI Citability;
- Brand Authority;
- Platform Readiness;
- conteúdo visível sem JS;
- H1;

- imagens/alt;
- schemas;
- SSR/pré-renderização;
- robots;
- crawlers;
- `llms.txt`;
- E-E-A-T;
- entidade/ `sameAs` ;
- fontes reconhecidas;
- citabilidade factual.

Nem todos os itens são regra universal.

67. Checklist AEO/GEO real em projeto

Daniel P possui checklist interno agrupado em:

- AEO;
- GEO;
- SEO;
- E-E-A-T.

Exemplos:

AEO

- FAQ;
- bloco “Em resumo”;
- TL;DR;
- headings interrogativos.

GEO

- entidades;
- variações semânticas;
- bloco “Quem deve ler/agir”;
- links internos.

SEO

- tamanho de title;
- meta description;
- keyword;
- contagem de palavras;
- headings.

E-E-A-T

- revisor;
 - data de revisão.
-

68. Limite do checklist Daniel

REGRA DE CONTENÇÃO

Esse checklist é uma implementação real de um projeto.

Ele não deve ser transformado automaticamente em checklist universal WSS nesta documentação AS-IS.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XV — NORMA TÉCNICA VIGENTE DE SEO/AEO/GEO

69. Camada normativa separada

A autoridade canônica vigente é:

- `WSS_Padrao_Tecnico_Operacional_SEO_AEO_GEO_v0.2_FINAL.md` — **padrão canônico vigente**.

O antigo `WSS_Padrao_Tecnico_Arquitetura_SEO_v0.1.md` está superado como autoridade autônoma e permanece somente como baseline histórico/proveniência.

O checklist AEO/GEO do Daniel P continua sendo um **caso real AS-IS**, não o checklist universal WSS. A partir do Gate 7, porém, a WSS possui matriz e checklists normativos próprios no caderno de anexos do v0.2 FINAL.

Referências operacionais principais para este Manual:

- `WSS_SEO_AEO_GEO_Matriz_Requisitos_Tecnicos_Parte2_v0.3`;
- `WSS_SEO_AEO_GEO_Matriz_Schema_por_Tipo_Pagina_v0.3`;
- `WSS_SEO_AEO_GEO_Matriz_Crawlers_Diretivas_v0.2`;
- `WSS_SEO_AEO_GEO_Checklist_PrePublicacao_v0.3`.

Esses anexos têm propriedade sobre o detalhe técnico; este Manual não os duplica.

Requisitos normativos que afetam QA

Entre os requisitos/guardrails vigentes estão, conforme aplicabilidade:

- arquitetura SSR-first/híbrida e conteúdo principal no HTML inicial para páginas públicas indexáveis;
- metadata/head específico por rota;
- canonical/origem oficial, redirects/status reais e 404 real;
- sitemap XML e `robots.txt`;
- staging/preview não indexável e fora do sitemap oficial;
- structured data coerente e validado;
- áreas privadas fora do sitemap e com `noindex` quando aplicável;
- Google Search Console configurado na publicação.

Performance normativa

Metas de Core Web Vitals são **recomendadas**, no p75 quando houver dados adequados:

- LCP $\leq 2,5$ s;
- INP ≤ 200 ms;
- CLS $\leq 0,1$.

Priorizar dados de campo/CrUX quando disponíveis; Lighthouse/PSI são diagnósticos e não garantia de ranking.

Decisões preservadas

- GA4: **recomendado, não obrigatório universal**;
- Bing Webmaster Tools: recomendado;

- ferramentas pagas: sem seleção definitiva;
- `llms.txt`: não é requisito universal;
- Search crawling e treinamento devem ser tratados separadamente, mas o default WSS para crawlers de treinamento permanece pendente;
- entrega padrão × recorrência permanece decisão estratégica/comercial pendente.

70. Regra obrigatória de separação

Este Manual deve mostrar:

AS-IS

O que o Dev e os projetos realmente fazem hoje.

NORMA TÉCNICA VIGENTE

O que deve ser cumprido por exigência aprovada.

A norma pode ser mais exigente que a rotina histórica. Ela não deve ser usada para afirmar que o Dev sempre executou historicamente o mesmo checklist.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XVI — SEGURANÇA AS-IS

71. Como segurança aparece hoje

O Dev não declarou executar um checklist manual isolado de segurança antes de toda publicação.

Segurança aparece principalmente em:

- auditorias gerais;
 - Lovable;
 - Codex;
 - hardening específico;
 - regressões.
-

72. Hardening real da Base Admin

A VAL-003 confirmou verificações reais de:

- credenciais fixas;
 - endpoints privilegiados;
 - papéis;
 - usuário sem papel;
 - anon ;
 - escalonamento;
 - RLS;
 - PII;
 - payloads;
 - autorização server-side;
 - secrets;
 - superfície pública;
 - resíduos de QA;
 - build/typecheck.
-

73. Papéis

Foram explicitamente testados:

- admin_webstar ;
- admin_equipe ;
- autenticado sem papel;
- anon .

Objetivos:

- Webstar mantém acesso previsto;
- Equipe não ganha acesso Webstar-only;
- sem papel não ganha acesso administrativo;
- anon não ganha acesso administrativo;
- escalonamento é bloqueado.

74. UI não basta

A auditoria verificou autorização **server-side**, não apenas ocultação visual.

Portanto:

botão escondido não é prova suficiente de controle de acesso.

75. RLS

Houve testes negativos de RLS em tabelas reais.

O objetivo era distinguir:

- leitura pública legítima;
- dados administrativos/PII que não devem ser públicos.

76. PII

Exemplos de superfícies que não devem ser expostas a anon incluem:

- Leads;
- configurações administrativas;
- auditorias;
- knowledge administrativa;
- dados sensíveis/caches externos conforme o caso.

77. Superfície pública legítima

Segurança não significa bloquear tudo.

Foram preservados/validados:

- tracking;
- chat;
- FAQ;
- formulários;
- páginas públicas;
- robots;
- sitemap.

78. Secrets

Em projeto real foram registrados como server-only:

- `SUPABASE_URL` ;
- `SUPABASE_PUBLISHABLE_KEY` ;
- `SUPABASE_SERVICE_ROLE_KEY` ;
- `INBOUND_WEBHOOK_SECRET` ;
- `LOVABLE_API_KEY` .

Limite

A lista é exemplo real, não inventário universal de todos os projetos.

79. Exemplo real de hardening

Foi encontrado endpoint histórico de seed de administradores com:

- credenciais fixas;
- acesso privilegiado.

Antes da remoção:

1. investigou-se uso/dependência;
2. confirmou-se ausência de consumidor produtivo;
3. endpoint foi neutralizado;
4. credenciais/acesso foram removidos;
5. regressões foram testadas.

80. Padrão de segurança observado

```
achado potencial
→ provar uso/dependências
→ medir risco
→ corrigir somente se seguro
→ testar regressão
```

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XVII — VALIDAÇÃO DE SEGURANÇA PROPORCIONAL

81. Procedimento reproduzível sustentado

Quando uma validação de segurança é necessária, o AS-IS técnico sustenta:

```
inventariar papéis/rotas/endpoints/server functions/tabelas
→ procurar credenciais e superfícies privilegiadas
→ provar dependências antes de remover
→ testar papéis
→ testar sem papel
→ testar anon
→ testar escalonamento
→ testar RLS negativa
→ confirmar superfície pública legítima
→ verificar PII
→ typecheck/build quando aplicável
→ revalidar regressões
```

82. Por que não vira SOP próprio agora

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não há evidência de que essa sequência seja um **gate universal obrigatório antes de toda publicação**.

Ela é uma prática técnica real/proporcional sustentada por auditorias.

Para evitar TO-BE:

- fica incorporada ao M6;
- não vira SOP independente nesta primeira versão.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XVIII — FEEDBACK E RODADAS DE CORREÇÃO

83. Canal predominante

O feedback chega predominantemente por:

WhatsApp

O Flow existe para tarefas/registros internos, mas o WhatsApp foi identificado como principal canal de feedback no AS-IS.

84. Formatos

Feedback pode chegar como:

- texto;
- áudio;
- mensagem;
- documento;
- prints;
- conversa.

Quando vem organizado em documento/prints, o Dev considera o processo mais simples.

85. Consolidação

O Dev costuma ler e iniciar correções.

Felipe/gestão podem consolidar quando necessário.

86. Quando voltar para gestão

Se a solicitação:

- está pouco clara;
- é contraditória;
- representa possível mudança de escopo;
- sugere nova funcionalidade;

o Dev valida com Felipe/gestão antes de executar.

87. ajustes.md

O arquivo centraliza a rodada.

Fluxo:

1. registrar pedidos;
 2. trabalhar item por item;
 3. marcar concluídos;
 4. pedir à IA revisão da lista completa;
 5. revalidar áreas afetadas;
 6. encerrar quando todos os itens foram tratados.
-

88. Antes/depois

Quando necessário, o Dev pode gerar documento com:

- prints anteriores;
- prints posteriores;
- comprovação das alterações.

Não é etapa universal obrigatória.

89. Relação com regressão

Cada item resolvido pode afetar outra área.

Por isso:

- não basta marcar como concluído;
 - é necessário revalidar área afetada;
 - a revisão final da lista ajuda a detectar resíduo.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XIX — AUSÊNCIAS ATUAIS DE PADRONIZAÇÃO

90. Lista consolidada deste Manual

O AS-IS não possui:

- suíte automatizada universal obrigatória;
- mesma bateria de testes em todos os projetos;
- método formal universal de root cause analysis;
- matriz sistemática de todos os viewports intermediários;
- checklist sistemático de estados vazios em todos os projetos;
- checklist sistemático de 404 em todos os projetos;
- smoke build universal comprovado para todos os sites;
- score SEOmator universal;
- uso obrigatório de SEOmator em toda entrega;
- mesmos thresholds AEO/GEO em todos os sites;
- checklist manual de segurança isolado antes de toda publicação;
- mesma matriz de RLS em todos os projetos;
- SOP universal de segurança antes de publicação;
- processo formal único de regressão;
- canal único obrigatório para feedback.

Essas ausências são achados e devem permanecer visíveis.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XX — OPORTUNIDADES FUTURAS — FORA DO ESCOPO

91. Oportunidades identificáveis

Sem desenhar solução:

- ampliar automação de testes;
- formalizar gates de build;
- padronizar regressão;
- formalizar segurança por criticidade;
- estruturar feedback prioritariamente em Flow;
- ampliar cobertura de estados de erro;
- formalizar acessibilidade.

Nenhuma dessas oportunidades deve reescrever o AS-IS.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXI — RELAÇÃO COM OS OUTROS MANUAIS

92. Relação com o Manual 3

M3 constrói:

- visual;
- responsividade;
- performance percebida.

M6 valida:

- resultado;
 - regressão;
 - comportamento.
-

93. Relação com o Manual 4

M4 explica capacidades.

M6 testa:

- Leads;
 - Chat;
 - Chatbot;
 - perfis;
 - Admin.
-

94. Relação com o Manual 5

M5 explica execução e sincronização.

M6 explica como validar alterações e diagnosticar quando algo falha.

95. Relação com o Manual 7

M6 é proprietário do método geral de QA.

M7 é proprietário do QA aplicado em produção/pós-publicação.

96. Relação com o Manual 8

M6 ensina:

como investigar.

M8 registra:

quais problemas conhecidos já ocorreram.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXII — SOPs DERIVADOS

97. SOP-008 — Executar QA Visual, Funcional e Responsivo

Este SOP formaliza:

- momento;
- preparação;
- desktop/notebook/mobile;
- páginas;
- Admin;
- Leads;
- Chat;
- perfis;
- evidências;
- stop condition.

98. SOP-009 — Diagnosticar e Corrigir Bug por Camada

Este SOP formaliza:

```
reproduzir  
→ localizar camada  
→ coletar evidência  
→ formular hipótese  
→ corrigir  
→ revalidar
```

Sem inventar método formal de causa raiz.

99. SOP-010 — Tratar Auditoria SEO, AEO e GEO

Este SOP formaliza:

```
receber auditoria  
→ comparar com estado atual  
→ classificar achado  
→ identificar falso positivo/limitação  
→ corrigir delta real  
→ validar
```

100. SOP-011 — Processar Rodada de Feedback via ajustes.md

Este SOP formaliza:

- receber;
 - consolidar;
 - resolver ambiguidade;
 - registrar;
 - executar;
 - marcar;
 - revisar lista;
 - revalidar.
-

101. Procedimentos incorporados, sem SOP próprio

Nesta primeira versão ficam incorporados ao M6:

- smoke build;
- validação de segurança proporcional.

Motivo:

- são práticas/procedimentos reais;
 - mas não foram comprovados como gate universal AS-IS.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXII — CASOS REAIS UTILIZADOS

102. Guilherme Mendonça

Uso no M6:

- QA corrente;
 - SEO/AEO/GEO;
 - build de produção;
 - smoke test;
 - validação de metadados.
-

103. Daniel P

Uso:

- CTA 404;
 - SEOmator;
 - falso positivo/visibilidade CSR;
 - acessibilidade pontual.
-

104. Mirela Albuquerque

Uso:

- debugging por ambiente;
 - Supabase;
 - console;
 - não reescrever Blog antes de verificar configuração.
-

105. Guilherme Maia

Uso:

- regressão em portabilidade;
 - preservar capacidade local;
 - validar delta adaptado.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXIV — CRITÉRIO DE CONCLUSÃO

106. Critério de conclusão deste Manual

Este Manual cumpre seu objetivo quando o leitor consegue responder:

1. quando ocorre o principal QA;
2. como Dev e Codex se dividem;
3. quais dispositivos são priorizados;
4. como o site público é percorrido;
5. como o Admin é testado;
6. como Leads/Chat/Chatbot entram no QA;
7. o que não é testado sistematicamente;
8. como o debugging começa;
9. quando usar console;
10. como escolher Codex/Claude/Lovable por camada;
11. como a causa é considerada confirmada no AS-IS;
12. como executar regressão contextual;
13. por que dev não garante build de produção;
14. como smoke build entra sem virar gate universal;
15. como uma auditoria SEO é triada;
16. como usar SEOMator sem obedecer cegamente;
17. quais dimensões AEO/GEO aparecem em casos reais;
18. como segurança é validada em auditorias reais;
19. como papéis/RLS/secrets/PII entram;
20. por que a validação de segurança permanece proporcional;
21. como feedback vira `ajustes.md`;
22. quais ausências atuais não devem ser silenciosamente formalizadas.

107. Fontes principais desta versão

Baseline canônico

1. `WSS_Arqueologia_Tecnica_ASIS_v0.3.md`;
2. `WSS_Matriz_Cobertura_ASIS_P0_P1_P2_v0.3_FINAL.md`;
3. `WSS_Handoff_Consolidacao_Manuais_SOPs_Skills_v0.2.md`;
4. `WSS_Metodo_Desenvolvimento_Sites_Plano_Conducao_v0.3.md`;
5. `WSS_Metodo_Desenvolvimento_Sites_Indice_Mestre_v0.1_DRAFT.md`.

Evidência técnica complementar

- `WSS_VAL_003_Auditorias_SEO_AEO_GEO_Seguranca_v0.1.md`;
- `WSS_VAL_004_Troubleshooting_Estudos_de_Caso_v0.1.md`.

Manuais já estabilizados usados para preservar fronteiras

- WSS_Manual_01_Visao_Geral_Toolchain_v1.1_FINAL.md ;
- WSS_Manual_02_Entrada_Contexto_Projeto_v1.1_FINAL.md ;
- WSS_Manual_03_Construcao_Visual_v1.1_FINAL.md ;
- WSS_Manual_04_Projeto_Base_Admin_Funcionalidades_v1.1_FINAL.md ;
- WSS_Manual_05_Development_Codex_Lovable_GitHub_v1.1_FINAL.md .

Norma técnica relacionada

- WSS_Padiao_Tecnico_Operacional_SEO_AEO_GEO_v0.2_FINAL.md — **padrão canônico vigente;**
- WSS_Padiao_Tecnico_Arquitetura_SEO_v0.1.md — baseline histórico/proveniência; **não é autoridade autônoma vigente;**
- WSS_SEO_AEO_GEO_Matriz_Requisitos_Tecnicos_Parte2_v0.3 ;
- WSS_SEO_AEO_GEO_Matriz_Schema_por_Tipo_Pagina_v0.3 ;
- WSS_SEO_AEO_GEO_Matriz_Crawlers_Diretivas_v0.2 ;
- WSS_SEO_AEO_GEO_Checklist_PrePublicacao_v0.3 .

Versões anteriores/superadas de AS-IS, Matriz, Handoff e Plano não foram usadas como baseline ativo.

Status: FINAL — APROVADO