

PROCEDIMENTO OPERACIONAL

MÉTODO DE DESENVOLVIMENTO DE SITES

SOP-011

Processar Rodada de Feedback via ajustes.md

VERSÃO

v1.1 FINAL

DATA

11/09/2026

STATUS

FINAL — APROVADO

PROCEDIMENTO OPERACIONAL

SOP-011

Processar Rodada de Feedback via ajustes.md

VERSÃO	v1.1 FINAL
DATA	11/09/2026
STATUS	FINAL — APROVADO
MANUAL PROPRIETÁRIO	Manual 6 — Testes, QA, Debugging, Segurança e SEO
MANUAIS RELACIONADOS	Manual 2 — Entrada e Contexto do Projeto; Manual 3 — Construção Visual; Manual 5 — Desenvolvimento com Codex, Lovable e GitHub; Manual 7 — Publicação, Entrega e Pós-Publicação; Manual 8 — Boas Práticas, Anti-Patterns, Práticas Depreciadas e Casos Reais
ESCOPO	AS-IS — recebimento, consolidação, execução e encerramento de rodadas de feedback/correção

NAVEGAÇÃO DO DOCUMENTO

Sumário

Estrutura do documento canônico v1.1 FINAL.

1. Finalidade	7
2. Gatilho — quando usar	7
3. Quando NÃO usar	7
4. Papéis	8
5. Pré-condições	9
6. Inputs	9
PARTE I — RECEBER O FEEDBACK	11
7. Canal predominante AS-IS	12
8. Passo 1 — Reunir todas as mensagens da rodada	12
9. Passo 2 — Identificar a versão do site à qual o feedback se refere	12
PARTE II — CONSOLIDAR OS PEDIDOS	14
10. Passo 3 — Transformar mensagens em itens claros	15
11. Passo 4 — Agrupar mensagens que tratam do mesmo problema	15
12. Passo 5 — Separar pedidos independentes	15
PARTE III — CLASSIFICAR O PEDIDO	17
13. Passo 6 — Classificar cada item	18
14. Ajuste claro	18
15. Pedido ambíguo	18
16. Contradição	19
17. Possível mudança de escopo	19
18. Bug	19
19. Conteúdo/fato pendente	19
PARTE IV — REGISTRAR EM AJUSTES.MD	21
20. Passo 7 — Atualizar ajustes.md	22
21. Estrutura mínima do item	22
22. Não inventar campos obrigatórios	22
23. Passo 8 — Manter separados os itens bloqueados	23

PARTE V — PREPARAR A EXECUÇÃO	24
24. Passo 9 — Confirmar contexto antes de alterar	25
25. Passo 10 — Definir escopo da alteração para a IA	25
26. Regra de contenção	25
PARTE VI — EXECUTAR ITEM POR ITEM	26
27. Passo 11 — Selecionar o próximo item executável	27
28. Passo 12 — Implementar a alteração	27
29. Passo 13 — Validar o item imediatamente	27
30. Passo 14 — Marcar item concluído	28
PARTE VII — ORDEM DA RODADA	29
31. AUSÊNCIA ATUAL DE PADRONIZAÇÃO — prioridade formal	30
32. Critério prático	30
PARTE VIII — EVITAR REGRESSÃO	31
33. Passo 15 — Preservar o que já foi aprovado	32
34. Passo 16 — Revalidar componente compartilhado quando alterado	32
35. Passo 17 — Revalidar breakpoint relacionado	32
36. Passo 18 — Revalidar fluxo funcional quando necessário	32
PARTE IX — IA CONFERINDO A LISTA	34
37. Passo 19 — Pedir revisão da lista inteira	35
38. Limite da revisão por IA	35
PARTE X — REVISÃO HUMANA FINAL DA RODADA	36
39. Passo 20 — Percorrer os itens concluídos	37
40. Passo 21 — Conferir itens ainda abertos	37
41. Passo 22 — Revalidar áreas afetadas	37
PARTE XI — EVIDÊNCIAS DE ANTES/DEPOIS	38
42. Uso opcional	39
43. AUSÊNCIA ATUAL DE PADRONIZAÇÃO	39
PARTE XII — NOVO FEEDBACK DURANTE A RODADA	40
44. Passo 23 — Registrar pedidos novos	41
45. Evitar rodada infinita silenciosa	41
PARTE XIII — FEEDBACK APÓS PUBLICAÇÃO TEMPORÁRIA	42

46. Contexto	43
47. Não confundir publicação temporária com entrega final	43
PARTE XIV — RELAÇÃO COM QA	44
48. Feedback não substitui QA	45
49. QA não substitui feedback	45
PARTE XV — RELAÇÃO COM DEBUGGING	46
50. Quando um pedido revela bug	47
PARTE XVI — RELAÇÃO COM ESCOPO	48
51. Sinais de possível mudança de escopo	49
52. Passo 24 — Interromper execução automática	49
53. Não mascarar feature nova como “ajuste”	49
PARTE XVII — CONTEÚDO E FATOS	50
54. Passo 25 — Separar preferência textual de fato	51
55. IA não completa lacuna factual	51
PARTE XVIII — STOP CONDITION	52
56. Condição de parada da rodada	53
57. Checklist mínimo	53
58. O que NÃO é necessário para encerrar	53
PARTE XIX — QUANDO NÃO ENCERRAR	55
59. Item marcado sem validação	56
60. Item ambíguo executado por inferência	56
61. Mudança de escopo executada sem decisão	56
62. Regressão encontrada	56
63. Cliente ainda precisa fornecer fato essencial	56
PARTE XX — AUSÊNCIAS ATUAIS DE PADRONIZAÇÃO	57
64. Não inventar	58
PARTE XXI — NORMA TÉCNICA VIGENTE	59
65. Feedback não pode remover requisito normativo silenciosamente	60
66. Norma ≠ preferência	60
PARTE XXII — ANTI-PATTERNS	61
67. Corrigir diretamente da memória do WhatsApp	62

68. Executar todos os pedidos em um único prompt amplo	62
69. Marcar item assim que o código muda	62
70. Deixar itens ambíguos escondidos no meio da lista	62
71. Tratar feature nova como correção simples	62
72. Confiar apenas na conferência da IA	62
73. Fechar rodada sem reler a lista completa	62
PARTE XXIII — OUTPUT	64
74. Saída esperada	65
PARTE XXIV — PRÓXIMO PASSO	66
75. Encaminhamento	67
76. Referências	67

1. Finalidade

Transformar feedback recebido em uma rodada controlada de correções, reduzindo perda de pedidos, interpretação ambígua, regressão e encerramento prematuro.

Fluxo principal:

```
feedback recebido
→ reunir pedidos
→ consolidar quando necessário
→ separar ajuste de dúvida/escopo
→ registrar em ajustes.md
→ executar item por item
→ marcar concluído
→ revisar lista completa
→ revalidar áreas afetadas
→ confirmar pendências
→ encerrar rodada
```

Este SOP documenta o fluxo AS-IS atualmente praticado.

Ele não cria um novo sistema de tickets, não substitui o Flow e não transforma WhatsApp em ferramenta formal única de gestão.

2. Gatilho — quando usar

Usar quando houver uma rodada de feedback sobre o projeto, especialmente após:

- revisão interna;
- publicação temporária;
- revisão do cliente;
- reunião;
- troca de mensagens;
- rodada de QA;
- validação de conteúdo;
- entrega parcial.

Pode ser usado tanto para uma lista pequena quanto para uma rodada ampla.

3. Quando NÃO usar

Não usar como substituto de:

- definição de escopo;
- briefing;

- Sitemap;
- debugging técnico profundo;
- QA completo;
- auditoria SEO/AEO/GEO;
- publicação.

Encaminhamentos

- bug não trivial → SOP-009;
 - QA geral → SOP-008;
 - auditoria SEO/AEO/GEO → SOP-010;
 - produção/entrega → SOP-012.
-

4. Papéis

4.1 Cliente

Pode:

- apontar correções;
- sugerir melhorias;
- validar textos/imagens;
- solicitar alterações;
- indicar problemas percebidos.

O cliente não decide sozinho, por meio de uma mensagem ambígua, se algo passa a fazer parte do escopo técnico.

4.2 Desenvolvedor

No AS-IS, costuma:

- ler o feedback;
 - compreender o pedido;
 - registrar;
 - começar correções;
 - marcar itens;
 - revisar a rodada;
 - revalidar o site.
-

4.3 Felipe / Gestão WSS

Pode:

- consolidar pedidos;
- resolver ambiguidade;
- interpretar contradições;
- decidir possível mudança de escopo;
- orientar prioridade;
- validar dúvida comercial/operacional.

Regra

Se o pedido:

- é pouco claro;
- contradiz decisão anterior;
- sugere feature nova;
- muda escopo;

DECISÃO HUMANA NECESSÁRIA — retornar para Felipe/gestão antes de executar.

5. Pré-condições

Antes de iniciar:

1. feedback foi recebido;
2. projeto/versão correta estão identificados;
3. Dev consegue acessar os pedidos;
4. contexto do projeto continua disponível;
5. existe local para manter `ajustes.md`.

Importante

Não começar a executar uma rodada extensa apenas a partir de memória das mensagens.

6. Inputs

Podem incluir:

- mensagens de WhatsApp;

- documento do cliente;
 - comentários de reunião;
 - prints;
 - vídeos;
 - áudios transcritos/interpretados;
 - mensagens internas;
 - notas do Flow;
 - feedback de QA;
 - lista de correções anterior;
 - `ajustes.md` já existente.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE I — RECEBER O FEEDBACK

7. Canal predominante AS-IS

O principal canal de feedback atualmente identificado é:

WhatsApp

O Flow existe para:

- Tasks;
- registros internos;
- organização operacional.

Importante

Isso não significa:

“todo feedback deve obrigatoriamente chegar por WhatsApp”.

É apenas o canal predominante no AS-IS.

8. Passo 1 — Reunir todas as mensagens da rodada

Antes de executar amplamente:

- localizar mensagens;
- reunir pedidos relacionados;
- identificar anexos;
- identificar prints;
- identificar áudios;
- identificar contexto adicional.

Objetivo

Evitar corrigir apenas a última mensagem e esquecer pedidos anteriores da mesma rodada.

9. Passo 2 — Identificar a versão do site à qual o feedback se refere

Confirmar se o cliente avaliou:

- preview;

- domínio temporário;
- versão atual;
- versão anterior.

Se houver dúvida

Comparar com o estado atual antes de alterar.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE II — CONSOLIDAR OS PEDIDOS

10. Passo 3 — Transformar mensagens em itens claros

Cada solicitação deve virar um item que permita responder:

- onde;
- o que mudar;
- qual resultado esperado.

Exemplo

Mensagem:

“Essa parte aqui poderia ficar mais forte.”

Item ainda insuficiente.

É necessário entender:

- qual seção;
- o que significa “mais forte”;
- tipografia?
- contraste?
- composição?
- conteúdo?

11. Passo 4 — Agrupar mensagens que tratam do mesmo problema

Se várias mensagens descrevem a mesma alteração:

- consolidar em um único item;
- preservar nuances importantes.

Objetivo

Evitar executar duas vezes a mesma correção com formulações diferentes.

12. Passo 5 — Separar pedidos independentes

Não transformar uma rodada inteira em uma única instrução ampla.

Separar por:

- página;
 - seção;
 - módulo;
 - tipo de alteração.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE III — CLASSIFICAR O PEDIDO

13. Passo 6 — Classificar cada item

Categorias práticas:

A. Ajuste claro

Pedido compreensível e dentro do escopo.

B. Pedido ambíguo

Não há informação suficiente para executar com segurança.

C. Contradição

Conflita com:

- briefing;
- Sitemap;
- decisão anterior;
- outro pedido do mesmo cliente.

D. Possível mudança de escopo

Pode incluir:

- nova página;
- nova funcionalidade;
- novo módulo;
- integração não prevista;
- mudança estrutural significativa.

E. Bug

Comportamento não funciona conforme esperado.

F. Conteúdo/fato pendente

Exige informação do cliente.

14. Ajuste claro

Pode seguir para `ajustes.md`.

15. Pedido ambíguo

Não inferir silenciosamente.

Ação

- esclarecer;
 - ou retornar à gestão quando necessário.
-

16. Contradição

Não escolher automaticamente qual mensagem "vence".

Ação

DECISÃO HUMANA NECESSÁRIA.

17. Possível mudança de escopo

Não implementar como se fosse ajuste normal.

Ação

- registrar;
 - separar da rodada executável;
 - retornar a Felipe/gestão.
-

18. Bug

Se simples e evidente:

- pode ser corrigido na rodada.

Se não trivial:

- encaminhar ao SOP-009.
-

19. Conteúdo/fato pendente

Se a alteração depende de informação:

- não inventar;
- registrar a pendência;
- manter o item aberto ou bloqueado.

Especial atenção a:

- formação;
 - títulos;
 - especialidades;
 - procedimentos;
 - resultados;
 - informações médicas;
 - endereços;
 - contatos;
 - outros fatos sensíveis.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE IV — REGISTRAR EM `ajustes.md`

20. Passo 7 — Atualizar `ajustes.md`

Centralizar os pedidos executáveis em:

`ajustes.md`

Função

O arquivo funciona como:

- memória da rodada;
- checklist humano;
- contexto para IA;
- proteção contra esquecimento.

21. Estrutura mínima do item

O AS-IS não define template universal.

Mas o item precisa ser claro o suficiente para execução.

Exemplo funcional:

```
- [ ] Home – Hero: trocar a foto principal pela imagem X enviada pelo cliente.
```

Ou:

```
- [ ] Especialidade Y – remover a frase Z e manter o restante da seção.
```

22. Não inventar campos obrigatórios

AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Não existe template formal obrigatório com:

- prioridade numérica;
- severidade;
- responsável;
- prazo;
- story points;

- ID;
- categoria;
- evidência.

Esses campos podem existir quando úteis, mas não fazem parte do AS-IS universal.

23. Passo 8 — Manter separados os itens bloqueados

Itens que dependem de decisão podem permanecer registrados como:

- pendente;
- bloqueado;
- aguardando retorno.

Regra

Não marcá-los como concluídos apenas porque não serão executados naquela rodada.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE V — PREPARAR A EXECUÇÃO

24. Passo 9 — Confirmar contexto antes de alterar

Antes da rodada:

- reler `ajustes.md`;
- reler contexto relevante;
- confirmar página/módulo;
- preservar áreas já validadas.

25. Passo 10 — Definir escopo da alteração para a IA

Ao trabalhar com Codex, Claude ou Lovable, informar:

- item atual;
- página/seção;
- o que pode mudar;
- o que deve permanecer.

Exemplo

```
Corrija apenas o item 4 do ajustes.md.  
Trabalhe somente a seção Hero mobile.  
Preserve o desktop já aprovado.  
Não altere outras páginas nem o Admin.
```

26. Regra de contenção

Não enviar toda a lista para implementação ampla quando:

- há itens muito diferentes;
- existem decisões pendentes;
- o risco de regressão é alto.

Trabalhar item por item ou em grupos coerentes.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE VI — EXECUTAR ITEM POR ITEM

27. Passo 11 — Selecionar o próximo item executável

Escolher item que:

- está claro;
 - não está bloqueado;
 - não exige decisão de escopo.
-

28. Passo 12 — Implementar a alteração

Usar a ferramenta apropriada:

Visual

- Codex;
- Claude.

Backend/banco/configuração

No pós-criação, o Codex pode ser a primeira camada de investigação/condução. Conforme a tarefa:

- Codex pode executar diretamente o que couber;
- Lovable/Lovable Cloud podem ser usados diretamente pelo Dev ou operados/orquestrados pelo Codex quando a integração for necessária.

A criação inicial da estrutura herdada de backend/banco/auth continua pertencendo ao remix da Base no Lovable.

Conteúdo

- ferramenta de IA + validação humana/factual.

Bug

- SOP-009 se não trivial.
-

29. Passo 13 — Validar o item imediatamente

Depois da alteração:

- abrir área;
- conferir resultado;
- comparar com o pedido.

Não marcar como concluído antes de verificar.

30. Passo 14 — Marcar item concluído

Quando validado:

- [x] ...

ou mecanismo equivalente já usado no projeto.

Importante

"Código alterado" não equivale a "item concluído".

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE VII — ORDEM DA RODADA

31. AUSÊNCIA ATUAL DE PADRONIZAÇÃO — prioridade formal

Não existe regra universal de:

- primeiro visual;
 - depois conteúdo;
 - depois funcional;
 - ordem por página;
 - prioridade numérica.
-

32. Critério prático

O Dev pode ordenar considerando:

- dependência;
- facilidade;
- risco;
- página;
- coerência da alteração.

DECISÃO HUMANA NECESSÁRIA

Quando ordem impactar:

- escopo;
 - entrega;
 - prazo;
 - prioridade comercial.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE VII — EVITAR REGRESSÃO

33. Passo 15 — Preservar o que já foi aprovado

Cada prompt/correção deve limitar:

- página;
 - seção;
 - breakpoint;
 - módulo.
-

34. Passo 16 — Revalidar componente compartilhado quando alterado

Se o item toca:

- Header;
- Footer;
- CTA compartilhado;
- componente reutilizado;
- layout genérico;

verificar outras páginas relevantes.

35. Passo 17 — Revalidar breakpoint relacionado

Se ajuste é:

- mobile → conferir mobile e depois regressão básica desktop;
 - desktop → conferir desktop e comportamento mobile relacionado.
-

36. Passo 18 — Revalidar fluxo funcional quando necessário

Exemplos:

- CTA;
- formulário;
- Lead;
- Chat;
- Admin.

Se a correção toca o fluxo:

executar novamente o caminho relevante.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE IX — IA CONFERINDO A LISTA

37. Passo 19 — Pedir revisão da lista inteira

Depois de executar os itens:

- pedir à IA para reler `ajustes.md`;
- comparar itens marcados;
- apontar algum item não tratado ou incompleto.

Objetivo

Usar a IA como segunda leitura da rodada.

38. Limite da revisão por IA

A IA não decide sozinha se:

- pedido subjetivo foi atendido;
- cliente ficará satisfeito;
- mudança de escopo foi aprovada;
- fato sensível está correto.

A revisão automatizada é auxiliar.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE X — REVISÃO HUMANA FINAL DA RODADA

39. Passo 20 — Percorrer os itens concluídos

O Dev deve:

- revisar a lista;
 - abrir áreas alteradas;
 - confirmar resultado.
-

40. Passo 21 — Conferir itens ainda abertos

Classificar cada restante como:

- ainda não executado;
 - bloqueado;
 - aguardando cliente;
 - aguardando gestão;
 - fora do escopo;
 - novo bug.
-

41. Passo 22 — Revalidar áreas afetadas

A profundidade depende da alteração.

Visual

- página;
- desktop/notebook/mobile relevantes.

Funcional

- fluxo;
- persistência;
- Admin.

Compartilhado

- demais superfícies que reutilizam o componente.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XI — EVIDÊNCIAS DE ANTES/ DEPOIS

42. Uso opcional

Quando necessário, o Dev pode gerar:

- prints de antes/depois;
- documento demonstrando alterações.

Finalidade

- comprovar correção;
 - facilitar comunicação com o cliente;
 - esclarecer alteração visual.
-

43. AUSÊNCIA ATUAL DE PADRONIZAÇÃO

Prints antes/depois:

- não são obrigatórios em toda rodada;
- não possuem formato universal.

São usados quando ajudam.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XII — NOVO FEEDBACK DURANTE A RODADA

44. Passo 23 — Registrar pedidos novos

Se chegam novos pedidos durante a execução:

- não depender da memória;
 - adicionar à lista;
 - classificar.
-

45. Evitar rodada infinita silenciosa

Se a lista muda continuamente:

- manter claro o que pertence à rodada original;
 - identificar novos pedidos;
 - escalar quando houver impacto de escopo/prioridade.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XIII — FEEDBACK APÓS PUBLICAÇÃO TEMPORÁRIA

46. Contexto

O site pode estar publicado em:

`webstarstudio.site`

para revisão do cliente.

Nesse momento:

- feedback chega;
 - Admin pode continuar sendo configurado em paralelo;
 - `ajustes.md` centraliza a rodada.
-

47. Não confundir publicação temporária com entrega final

A existência de URL pública não impede:

- ajustes;
 - refinamento;
 - correções;
 - validação final.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XIV — RELAÇÃO COM QA

48. Feedback não substitui QA

O cliente pode não detectar:

- rota quebrada;
- persistência de Lead;
- problema de perfil;
- regressão técnica;
- SEO;
- erro de produção.

O SOP-008 continua necessário conforme o estágio.

49. QA não substitui feedback

Uma experiência tecnicamente correta pode ainda precisar de:

- conteúdo;
- imagem;
- tom;
- ajuste visual;
- preferência do cliente.

As duas camadas são complementares.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XV — RELAÇÃO COM DEBUGGING

50. Quando um pedido revela bug

Exemplo:

"O formulário diz que enviou, mas não recebemos nada."

Isso não deve ser tratado apenas como alteração visual.

Ação

- reproduzir;
 - usar SOP-009;
 - depois voltar ao item em `ajustes.md`.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XVI — RELAÇÃO COM ESCOPO

51. Sinais de possível mudança de escopo

Exemplos:

- nova página não prevista;
 - nova integração;
 - novo módulo;
 - nova jornada;
 - nova funcionalidade;
 - alteração estrutural ampla.
-

52. Passo 24 — Interromper execução automática

Quando houver sinal de escopo:

1. registrar o pedido;
 2. não implementar;
 3. retornar a Felipe/gestão;
 4. aguardar decisão.
-

53. Não mascarar feature nova como “ajuste”

Essa separação protege:

- contrato;
 - prazo;
 - arquitetura;
 - expectativa do cliente.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XVII — CONTEÚDO E FATOS

54. Passo 25 — Separar preferência textual de fato

Preferência

Exemplo:

- "quero um texto mais direto".

Pode ser reescrito.

Fato

Exemplo:

- formação;
- título;
- cidade de atendimento;
- procedimento realizado.

Precisa ser confirmado.

55. IA não completa lacuna factual

Se o feedback pede adicionar dado inexistente:

- solicitar fonte;
 - usar material oficial;
 - não inferir.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XVIII — STOP CONDITION

56. Condição de parada da rodada

O SOP está concluído quando:

todos os pedidos recebidos para a rodada foram registrados e classificados; os itens executáveis foram implementados e marcados; a lista completa foi revisada; as áreas afetadas foram revalidadas; e qualquer item remanescente está explicitamente identificado como pendente, bloqueado, aguardando decisão ou fora do escopo.

57. Checklist mínimo

Antes de encerrar:

- ☐ feedback da rodada reunido;
- ☐ versão avaliada identificada;
- ☐ pedidos consolidados;
- ☐ itens ambíguos separados;
- ☐ contradições identificadas;
- ☐ possíveis mudanças de escopo escaladas;
- ☐ itens executáveis registrados em `ajustes.md`;
- ☐ cada item executado foi validado;
- ☐ itens concluídos marcados;
- ☐ IA realizou conferência da lista completa quando útil;
- ☐ Dev revisou a lista;
- ☐ áreas afetadas revalidadas;
- ☐ componentes compartilhados revalidados quando necessário;
- ☐ pendências remanescentes classificadas;
- ☐ fatos sensíveis não foram inventados.

58. O que NÃO é necessário para encerrar

Não exigir universalmente:

- cliente responder novamente a cada item;
- documento formal de aprovação por item;
- print antes/depois em toda correção;
- ticket no Flow para cada ajuste;
- prioridade/severidade numérica;
- template rígido de `ajustes.md`;

- reunião após toda rodada.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XIX — QUANDO NÃO ENCERRAR

59. Item marcado sem validação

Não encerrar.

Reabrir o item.

60. Item ambíguo executado por inferência

Revisar com humano.

61. Mudança de escopo executada sem decisão

Não considerar a rodada corretamente concluída.

Registrar e retornar à gestão.

62. Regressão encontrada

Encaminhar:

- SOP-008 para QA;
- SOP-009 para bug;

conforme o caso.

63. Cliente ainda precisa fornecer fato essencial

Manter pendência explícita.

Não inventar conclusão.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XX — AUSÊNCIAS ATUAIS DE PADRONIZAÇÃO

64. Não inventar

O AS-IS não possui:

- canal único obrigatório de feedback;
 - template universal de `ajustes.md`;
 - IDs obrigatórios;
 - severidade numérica;
 - prioridade universal;
 - workflow formal de aprovação por item;
 - SLA universal para correções;
 - obrigação de registrar todo pedido no Flow;
 - documento antes/depois obrigatório;
 - ordem rígida de execução;
 - aceite formal do cliente item por item.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXI — NORMA TÉCNICA VIGENTE

65. Feedback não pode remover requisito normativo silenciosamente

Se um pedido do cliente contradiz:

- segurança;
- arquitetura pública;
- SEO;
- autenticação;
- privacidade;
- outra norma vigente;

não executar automaticamente.

Quando o conflito envolver SEO/AEO/GEO, consultar:

- WSS_Padrao_Tecnico_Operacional_SEO_AEO_GEO_v0.2_FINAL .

Ação

DECISÃO HUMANA NECESSÁRIA, preservando a camada normativa aplicável.

66. Norma ≠ preferência

É possível adaptar:

- visual;
- linguagem;
- composição.

Não é aceitável tratar requisito técnico aprovado como simples preferência visual.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXII — ANTI-PATTERNS

67. Corrigir diretamente da memória do WhatsApp

Risco:

- esquecer pedido;
 - misturar rodadas;
 - interpretar errado.
-

68. Executar todos os pedidos em um único prompt amplo

Risco:

- regressão;
 - alteração fora de escopo;
 - dificuldade de validar item a item.
-

69. Marcar item assim que o código muda

O item só termina após validação.

70. Deixar itens ambíguos escondidos no meio da lista

Separar para decisão.

71. Tratar feature nova como correção simples

Risco de escopo.

72. Confiar apenas na conferência da IA

A revisão humana continua necessária.

73. Fechar rodada sem reler a lista completa

Esse é exatamente o risco que `ajustes.md` busca reduzir.

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXIII — OUTPUT

74. Saída esperada

Ao final:

Registro

- `ajustes.md` atualizado;
- itens concluídos marcados;
- pendências explicitadas.

Execução

- correções implementadas;
- mudanças fora de escopo não executadas silenciosamente.

Validação

- áreas afetadas revalidadas;
- regressões tratadas/encaminhadas.

Governança

- dúvidas/contradições/escalonamentos resolvidos ou registrados;
 - fatos pendentes preservados como pendências.
-

WSS · MÉTODO DE DESENVOLVIMENTO DE SITES

PARTE XXIV — PRÓXIMO PASSO

75. Encaminhamento

Depois do SOP-011:

Se ainda houver QA

→ SOP-008 — Executar QA Visual, Funcional e Responsivo

Se houver bug

→ SOP-009 — Diagnosticar e Corrigir Bug por Camada

Se houver auditoria técnica

→ SOP-010 — Tratar Auditoria SEO, AEO e GEO

Se a rodada está encerrada e o projeto está pronto

→ SOP-012 — Preparar, Publicar, Validar Produção e Entregar Admin

76. Referências

Manual proprietário

- WSS_Manual_06_Testes_QA_Debugging_Seguranca_SEO_v1.1_FINAL.md

Manuais relacionados

- WSS_Manual_02_Entrada_Contexto_Projeto_v1.1_FINAL.md ;
- WSS_Manual_03_Construcao_Visual_v1.1_FINAL.md ;
- WSS_Manual_05_Desenvolvimento_Codex_Lovable_GitHub_v1.1_FINAL.md ;
- WSS_Manual_07_Publicacao_Entrega_Pos_Publicacao_v1.1_FINAL.md ;
- WSS_Manual_08_Boas_Praticas_AntiPatterns_Depreciadas_Casos_Reais_v1.1_FINAL.md .

Baseline

- WSS_Arqueologia_Tecnica_ASIS_v0.3.md ;
- WSS_Matriz_Cobertura_ASIS_P0_P1_P2_v0.3_FINAL.md ;
- WSS_Handoff_Consolidacao_Manuais_SOPs_Skills_v0.2.md ;
- WSS_Metodo_Desenvolvimento_Sites_Plano_Conducao_v0.3.md ;
- WSS_Metodo_Desenvolvimento_Sites_Indice_Mestre_v0.1_DRAFT.md .

Contexto de Processo

- Roadmap_Sites_WSS_Modelo_Padrao_v0.2_FINAL.md .
-

Status: FINAL — APROVADO