WhatsApp Sender - Brainstorm de Melhorias

Roadmap de Evolução do Produto

---

1. MÉTRICAS E ANALYTICS

1.1 Métricas de Engajamento Individual

Métrica	Descrição	Fórmula/Lógica
Taxa de Leitura	Percentual de mensagens lidas por contato	`(readCount / sentCount) × 100`
Tempo até Leitura	Velocidade de resposta do contato	`lastReadAt - lastSentAt`
Score de Engajamento	Classificação de 1-5 estrelas	Baseado em taxa + frequência
Última Interação	Tempo desde último contato	`now() - lastReadAt`

1.2 Métricas de Campanha

Métrica	Descrição
Taxa de Entrega	% de mensagens entregues por batch
Taxa de Leitura por Template	Qual conteúdo performa melhor
Melhor Horário	Correlação hora × taxa de leitura
Conversão por Grupo	Engajamento por segmento de contatos

1.3 Métricas Globais (Dashboard)

- Total de mensagens enviadas (dia/semana/mês)
- Total de leituras confirmadas
- Taxa média de engajamento
- Contatos mais engajados (top 10)
- Contatos inativos (não leram nos últimos 30 dias)

1.4 Métricas em Tempo Real

- Polling metrics já implementado
- Fila de mensagens pendentes
- Velocidade de processamento
- Uptime do WhatsApp Web

---

2. MELHORIAS DE UI/UX

2.1 Design System

Melhoria	Impacto
Dark Mode	Conforto visual, tendência moderna
Glassmorphism	Visual premium e sofisticado
Micro-animações	Feedback sutil, UX fluida
Skeleton Loading	Percepção de velocidade
Sonner Toasts	Notificações não-intrusivas

2.2 Dashboard Analítico

- Cards de KPIs com sparklines
- Gráficos interativos (Recharts/Chart.js)
- Heatmap de horários de leitura
- Timeline de atividades recentes
- Comparativo semanal/mensal

2.3 Gestão de Contatos

- Filtros avançados (por grupo, engajamento, data)
- Busca com autocomplete e fuzzy search
- Seleção múltipla + ações em massa
- Tags coloridas personalizáveis
- Ordenação por múltiplos critérios
- Visualização em cards ou tabela
- Exportação para CSV/Excel

2.4 Editor de Mensagens

- Preview em tempo real do template
- Variáveis dinâmicas: `{nome}`, `{empresa}`, `{data}`
- Contador de caracteres
- Preview de mídia antes do envio
- Formatação rica (negrito, itálico, código)
- Biblioteca de emojis integrada
- Sugestões de templates populares

2.5 Experiência de Envio

- Barra de progresso animada
- Logs em tempo real com cores
- Estimativa de tempo restante
- Pausar/Retomar campanha
- Preview de próximo contato
- Confirmação visual de cada envio

---

3. NOVAS FUNCIONALIDADES

3.1 Automação Inteligente

Feature	Descrição	Prioridade
Campanhas Recorrentes	Envios automáticos semanais/mensais	Alta
Follow-up Automático	Reenvio para quem não leu após X dias	Alta
Segmentação Dinâmica	Enviar só para contatos engajados	Média
Horário Inteligente	Enviar no melhor horário por contato	Média
Limites de Frequência	Evitar spam para o mesmo contato	Alta

3.2 Gestão de Templates

- Biblioteca categorizada (vendas, suporte, marketing)
- Favoritos e templates mais usados
- Histórico de uso por template
- Duplicar e editar templates
- Templates com variáveis condicionais
- Templates multilíngua

3.3 Integrações

Integração	Benefício
Webhooks	Notificar sistemas externos de leituras
Google Sheets	Importação/exportação de contatos
Zapier/Make	Automações com outras ferramentas
HubSpot/Pipedrive	Sincronização com CRM
Calendário	Agendamento visual de campanhas

3.4 Relatórios e Exportação

- Relatórios em PDF com gráficos
- Exportação para Excel pivot-friendly
- Envio automático de relatórios por email
- Comparativo entre períodos
- A/B testing com análise estatística

3.5 Gestão de Grupos

- Grupos dinâmicos (baseados em regras)
- Merge e split de grupos
- Importação com mapeamento automático
- Detecção de duplicatas
- Validação de números (DDD, formato)

3.6 Central de Histórico

- Timeline de todas as mensagens enviadas por contato
- Status de cada mensagem (enviada, entregue, lida)
- Busca no histórico
- Filtro por período
- Anexos enviados

3.7 Relatórios Automáticos via WhatsApp

Objetivo: Enviar resumo das métricas para um número cadastrado após cada campanha.

📊 Relatório Imediato (ao finalizar envio)

Métrica	Descrição
Quantidade enviada	Total de mensagens
Template/promoção	Nome do conteúdo enviado
Horário início/fim	Ex: 18:00 - 18:12
Tempo de execução	Duração total
Velocidade média	Msgs/minuto
Erros/falhas	Quantidade e motivo
Contatos ignorados	Duplicados, blacklist
Comparativo	"15% mais rápido que ontem"

📈 Relatório de Engajamento (delayed)

Métrica	Descrição
Taxa de abertura	% que leu a mensagem
Tempo médio até leitura	"Média: 23 min"
Pico de leituras	"Maioria leu entre 19:00-20:00"
Leitores rápidos	Quem leu em <5min (VIPs)
Não lidos	Lista de quem não abriu
Respostas recebidas	Quantidade total
Taxa de resposta	% que respondeu
Palavras-chave	"pedido", "cardápio", "preço"
Novos contatos	Primeiro contato com eles
Top engajados	Ranking de interação

🗓️ Relatório Comparativo Semanal

- Tabela comparativa dia a dia
- Média de engajamento da semana
- Insights automáticos ("Terça teve melhor engajamento")
- Tendências de crescimento/queda

⚙️ Configurações de Horário

Opção	Exemplo
Horário fixo	Sempre às 23:00
X horas após envio	4 horas depois
Final do expediente	Configurável por dia

🎯 Personalizações Avançadas

- **Segmentação**: Relatório diferente por grupo (VIP vs novos)
- **Múltiplos destinatários**: Dono, gerentes, grupo de gestão
- **Formato visual**: Texto, imagem/infográfico, ou áudio (TTS)
- **Alertas inteligentes**: "⚠️ Taxa abaixo de 40%" ou "🎉 Recorde!"

---

4. FUNCIONALIDADES AVANÇADAS

4.1 Chatbot Básico

- Respostas automáticas para keywords
- Horário de atendimento
- Mensagem de ausência
- FAQ automatizado

4.2 Multiusuário

- Login com diferentes perfis
- Permissões por funcionalidade
- Auditoria de ações
- Sessões WhatsApp separadas

4.3 Multi-device

- Conectar múltiplos números WhatsApp
- Balanceamento de carga entre números
- Rotação automática para evitar bloqueios
- Dashboard unificado

4.4 Segurança e Compliance

- Opt-out automático (responder SAIR)
- Blacklist de contatos
- Rate limiting configurável
- Backup de dados

- LGPD compliance (exportar/deletar dados)

4.5 Background Service (Windows)

Objetivo: Executar o app automaticamente ao ligar o computador, sem precisar abrir o navegador.

Benefícios

- Mensagens agendadas são enviadas mesmo sem interação
- WhatsApp permanece conectado 24/7
- Não depende do navegador para funcionar

Implementação

Opção	Descrição	Dificuldade
Windows Service	Serviço nativo via `node-windows`	Média
PM2 + Startup	Gerenciador de processos Node.js	Fácil
Task Scheduler	Agendador nativo do Windows	Fácil

Componentes Necessários

- **Scheduler Service**: Verificar periodicamente mensagens agendadas
- **Health Check**: Reconectar WhatsApp se desconectar
- **System Tray Icon** (opcional): Ícone na bandeja com status

---

5. PRIORIZAÇÃO SUGERIDA

Fase 1: Quick Wins (1-2 semanas)

1. Dashboard com KPIs básicos
2. Filtros avançados na lista de contatos
3. Dark mode
4. Melhorias visuais (animações, skeleton)

Fase 2: Core Features (2-4 semanas)

1. Editor de templates com variáveis
2. Campanhas recorrentes
3. Follow-up automático
4. Relatórios básicos em PDF

Fase 3: Diferenciação (1-2 meses)

1. Segmentação inteligente
2. Horário inteligente de envio
3. Webhooks e integrações
4. Central de histórico

Fase 4: Escala (2-3 meses)

1. Multi-device

2. Multiusuário
3. Chatbot básico
4. API pública

---

6. STACK TÉCNICA RECOMENDADA

Área	Tecnologia
Gráficos	Recharts ou Tremor
PDF	@react-pdf/renderer ou jsPDF
Exportação	xlsx, papaparse
Notificações	Sonner (já instalado)
Animações	Framer Motion
Forms	React Hook Form + Zod
Tabelas	TanStack Table
Drag & Drop	dnd-kit

---

7. REFATORAÇÃO ARQUITETURAL

> ***Objetivo**: Distribuir funções e componentes para evitar aglomerado de código em arquivos únicos, melhorando testabilidade, manutenibilidade e reutilização.*

7.1 Mapeamento de Arquivos Críticos

Arquivo	Tamanho	Problema	Prioridade
`group-management-dialog.tsx`	12KB / 292 linhas	Lógica de negócio + UI misturados	Alta
`contact-engagement.tsx`	10KB / 227 linhas	Cálculos de engajamento inline	Alta
`status-panel.tsx`	8KB / 196 linhas	Estado complexo no componente	Média
`send-form.tsx`	8KB / 228 linhas	Muitas props (prop drilling)	Média
`schedule-item.tsx`	7KB / 138 linhas	Confirm inline, sem service	Baixa
`recipient-selector.tsx`	7KB / ~200 linhas	Lógica de seleção inline	Baixa

7.2 Tarefas de Extração - Hooks

[ ] 7.2.1 `useGroupManagement` Hook

Arquivo: `src/hooks/use-group-management.ts`
Extrair de: `group-management-dialog.tsx`

```
Estado:
├── actionState (move/remove)
├── isLoading
├── confirmRemove

Handlers:
├── handleRemove()
├── handleMove()
├── startMoveAction()
└── cancelAction()

Computed:
├── groupContacts
├── otherGroups
└── canMove
```

[ ] 7.2.2 `useEngagementStatus` Hook

Arquivo: `src/hooks/use-engagement-status.ts`
Extrair de: `contact-engagement.tsx`

```
Lógica:
├── getEngagementStatus(stats)
├── formatRelativeDate(date)
└── STATUS_CONFIG mapping

Retorno:
├── status
├── config (label, className, icon)
└── formattedDates
```

[ ] 7.2.3 `useSendForm` Hook

Arquivo: `src/hooks/use-send-form.ts`
Extrair de: Página `send/page.tsx` + `send-form.tsx`

```
Estado:
├── message, selectedFile
├── selectedGroupId
├── isScheduleMode, scheduleDate
├── selectedTemplateId

Computed:
├── recipientsCount
├── estimatedTime
├── canSubmit (validação)
```

7.3 Tarefas de Extração - Services

[ ] 7.3.1 `GroupService`

Arquivo: `src/lib/GroupService.ts`

```
// Operações de grupo centralizadas
moveContactToGroup(contactId, fromGroupId, toGroupId): Promise<void>
removeContactFromGroup(contactId, groupId): Promise<void>
getContactsInGroup(groupId): Contact[]
validateGroupOperation(contact, action): ValidationResult
```

[ ] 7.3.2 `EngagementService`

Arquivo: `src/lib/EngagementService.ts`

```
// Cálculos de engajamento
calculateEngagementRate(stats): number
getEngagementStatus(stats): EngagementStatus
getEngagementBadgeConfig(status): BadgeConfig
formatEngagementDate(date): string
```

7.4 Tarefas de Extração - Sub-componentes

[ ] 7.4.1 Quebrar `GroupManagementDialog`

Diretório: `src/components/contacts/group-management/`

```
├── index.ts                    # Re-exports
├── GroupManagementDialog.tsx   # Container principal
├── ContactRow.tsx              # Linha da tabela
├── MoveContactForm.tsx         # Select + OK/Cancel
├── RemoveConfirmDialog.tsx     # AlertDialog
└── types.ts                    # Interfaces
```

[ ] 7.4.2 Quebrar `StatusPanel`

Diretório: `src/components/send/status/`

```
├── index.ts
├── StatusPanel.tsx             # Container
├── ProgressSection.tsx         # Barra de progresso
├── LogsSection.tsx             # Lista de logs
├── SchedulesList.tsx           # Lista de agendamentos
└── StatusIndicator.tsx         # Badge de status
```

[ ] 7.4.3 Quebrar `ScheduleItem`

Diretório: `src/components/schedule/`

```
├── index.ts
├── ScheduleItem.tsx            # Container
├── ScheduleHeader.tsx          # Header expandível
├── ScheduleDetails.tsx         # Conteúdo expandido
├── ContactsPreview.tsx         # Preview de contatos
└── CancelConfirmDialog.tsx     # Confirmação
```

7.5 Ordem de Implementação Sugerida

> ⚠️ ***IMPORTANTE**: Implementar uma tarefa por vez para manter estabilidade.*

Fase A: Extrações de Lógica (Sem quebrar UI)

1. ☐ Criar `GroupService.ts`
2. ☐ Criar `EngagementService.ts`
3. ☐ Criar `useGroupManagement` hook
4. ☐ Criar `useEngagementStatus` hook
5. ☐ Criar `useSendForm` hook

Fase B: Refatorar Componentes Existentes

6. ☐ Refatorar `group-management-dialog.tsx` para usar hook + service
7. ☐ Refatorar `contact-engagement.tsx` para usar hook + service
8. ☐ Refatorar `send-form.tsx` para usar hook

Fase C: Quebrar Componentes (Opcional)

9. ☐ Quebrar `GroupManagementDialog` em sub-componentes
10. ☐ Quebrar `StatusPanel` em sub-componentes
11. ☐ Quebrar `ScheduleItem` em sub-componentes

7.6 Critérios de Conclusão

Cada tarefa está completa quando:

- ☐ Arquivo criado no local correto
- ☐ Tipos TypeScript definidos
- ☐ Imports atualizados no componente original
- ☐ Testes manuais passam (navegação funciona)
- ☐ Build sem erros (`pnpm build`)
- ☐ Nenhuma regressão visual

Documento gerado em: 05/02/2026 WhatsApp Sender - Roadmap v1.0

8. GAPS FUTUROS E ROADMAP

> ***Status**: Identificado em 09/02/2026 apos diagnostico geral da aplicacao.*

8.1 Gaps Criticos (Curto Prazo)

Gap	Problema	Acao Sugerida	Esforco
Prisma Schema	ContactWhereUniqueInput.number error	Verificar @@unique no schema	1h
Error Boundaries	Erros quebram toda a UI	Criar ErrorBoundary.tsx	2h
Loading States	Gerenciamento individual	Suspense boundaries	2h

---

8.2 Gaps de Arquitetura (Medio Prazo)

8.2.1 Quebrar whatsapp.ts (828 linhas)

Estrutura proposta:

- lib/whatsapp/WhatsAppClient.ts (Conexao e eventos)
- lib/whatsapp/MessageSender.ts (Envio de mensagens)
- lib/whatsapp/PollingService.ts (Polling de confirmacoes)
- lib/whatsapp/index.ts (Re-exports)

8.2.2 Zustand Slices Pattern

Estrutura proposta:

- store/contactsSlice.ts
- store/logsSlice.ts
- store/sendingSlice.ts
- store/index.ts

8.2.3 Validacao Zod nas APIs

14 rotas API sem schema validation - adicionar Zod schemas.

8.3 Gaps de Qualidade (Longo Prazo)

Area	Prioridade	Ferramenta
Testes unitarios (Services)	Alta	Jest
Testes de hooks	Media	@testing-library/react-hooks
Testes E2E	Baixa	Playwright

8.4 Ordem de Implementacao

Fase	Prioridade	Tarefa	Esforco
E1	Alta	Corrigir Prisma schema	1h
E2	Alta	Error Boundaries	2h
E3	Media	Quebrar whatsapp.ts	4h

E4	Media	Zustand slices	3h
E5	Media	Validacao Zod	3h
E6	Baixa	Testes unitarios	4h
E7	Baixa	Testes integracao	4h

Documento atualizado em: 09/02/2026 WhatsApp Sender - Roadmap v1.1