# Documento de Requisitos de Software (DRS)

**Projeto:** Sistema de Gestão de Atendimentos (Service Desk/Help Desk)  
**Integração de Chat:** WhatsApp via **Evolution API**  
**Autor:** —  
**Data:** 01/09/2025  
**Versão:** 1.0

## 1. Visão Geral

### 1.1 Objetivo

Projetar e implementar uma aplicação para gestão de atendimentos ao cliente que centralize tickets, comunicações (com foco em WhatsApp), SLA, indicadores customizáveis, automações e relatórios operacionais/gerenciais.

### 1.2 Escopo

* Abertura, triagem, atendimento, resolução e pós-atendimento de tickets.
* Omnichannel (foco inicial em WhatsApp), com expansão futura para e-mail e chat web.
* Dashboards e relatórios com **indicadores customizados** (construtor de métricas/visões).
* Integração **WhatsApp via Evolution API** (envio/recebimento, mídia, webhooks, status de entrega, sessões, fila/roteamento).
* Gestão de usuários, perfis, permissões e auditoria.
* Gestão de SLA, regras de prioridade e automações.
* LGPD, segurança, rastreabilidade e backups.

### 1.3 Stakeholders

* **Clientes finais** (solicitantes)
* **Atendentes** (nível 1 / 2 / 3)
* **Gestores de Suporte** (coordenação, indicadores, qualidade)
* **Administradores** (configuração, integrações, segurança)
* **TI/DevOps** (deploy, observabilidade, SRE)

### 1.4 Definições/Referências

* **Ticket**: unidade de atendimento que reúne conversas, atividades e SLA.
* **Evolution API**: camada de API para WhatsApp (sessões, mensagens, webhooks).
* **LGPD**: Lei Geral de Proteção de Dados.

## 2. Requisitos Funcionais (RF)

### RF-01 – Cadastro e Autenticação

1. Login por e-mail/senha e SSO (OIDC) opcional.
2. MFA opcional (TOTP).
3. Recuperação de senha (token expira em 60 min).

### RF-02 – Gestão de Usuários, Perfis e Permissões

1. Perfis: **Atendente**, **Gestor**, **Administrador**, **Cliente (Portal)**.
2. Permissões granularizadas por módulo/ação (RBAC), ex.: criar/editar/fechar ticket, ver dados sensíveis, configurar SLA.
3. Times/Filas: associação de usuários a times, com escalas/turnos.

### RF-03 – Cadastro de Clientes e Contatos

1. Entidade **Cliente** (empresa) e **Contato** (pessoa).
2. Identificação automática por número de WhatsApp.
3. Enriquecimento de dados: e-mail, CNPJ/CPF (opcional), segmento, SLA contratado.
4. Histórico de atendimentos por cliente/contato.

### RF-04 – Integração WhatsApp (Evolution API)

1. **Sessões**: gerenciar múltiplas sessões/instâncias (multiempresa/multinúmero).
2. **Recebimento** de mensagens via **webhook** (texto, imagem, áudio, documento, localização, contatos).
3. **Envio** de mensagens 1:1 a partir do ticket (texto, mídia, respostas rápidas).
4. **Sincronização de status**: enviado, entregue, lido, falha.
5. **Eventos**: message.upsert, message.status, presence, connection/update.
6. **Anti-duplicidade**: idempotência por message\_id.
7. **Fallback**: reenvio com backoff em caso de falha transitória.
8. **Bloqueio/opt-out**: marcar contato como opt-out; não enviar comunicações proativas.
9. **Templates** (mensagens reusáveis/boilerplate) internas ao sistema.
10. **Etiquetas** (tags) no WhatsApp espelhadas no ticket (opcional).

### RF-05 – Abertura e Triagem de Tickets

1. Geração automática de ticket ao receber primeira mensagem de um número desconhecido ou comando específico.
2. Abertura manual pelo atendente/gestor.
3. Campos: título, descrição, canal, cliente/contato, categoria, subcategoria, prioridade, **SLA aplicável**, anexos.
4. Sugestão de categoria por NLP (opcional, fase 2).
5. **De-para**: amarrar conversa do WhatsApp ao ticket.

### RF-06 – Roteamento e Atribuição

1. Atribuição manual a atendente/time.
2. Atribuição automática baseada em: fila, disponibilidade, skill, idioma, carga.
3. Transferência entre atendentes/times, preservando histórico.
4. Fila de espera com **tempo de espera estimado**.

### RF-07 – Atendimento Operacional (Console do Agente)

1. Visualização do **thread** de mensagens WhatsApp + notas internas.
2. Envio de mensagens (texto/mídia), snippets e templates.
3. Solicitar dados ao cliente (campos dinâmicos com validação).
4. Mencionar colegas (@) e anexar arquivos.
5. Alterar status do ticket: Aberto, Em atendimento, Pendente Cliente, Pendente Terceiros, Resolvido, Fechado, Cancelado.
6. **Macros** de atendimento (ações em lote: mudar status + enviar mensagem + aplicar tag).
7. **Satisfação (CSAT)**: disparo de pesquisa ao fechar.

### RF-08 – SLA e Regras

1. SLA por **prioridade + categoria + cliente/contrato**:
   * Tempo máximo de **primeira resposta**.
   * Tempo máximo de **resolução**.
2. Pausas de SLA (Pendente Cliente).
3. Alertas de violações iminentes; escalonamento (notificar gestor / mover fila).
4. Calendário de atendimento (horário útil, feriados).

### RF-09 – Base de Conhecimento (BK)

1. Artigos com categorias, níveis de acesso (público, interno).
2. Editor rich text, anexos, versões e revisão por pares.
3. Sugerir artigos automaticamente durante o atendimento (por categoria/palavra-chave).
4. Métricas: artigos mais usados, taxa de resolução por artigo.

### RF-10 – Relatórios e Indicadores Customizados

1. **Dashboards**: visão macro (volume, SLA, TMA, TME, backlog, CSAT, NPS\* opcional).
2. **Construtor de indicadores** (no/low-code):
   * Selecionar dataset (tickets, mensagens, atendentes, clientes).
   * Definir medidas (contagem, média, percentil, soma) e dimensões (tempo, categoria, atendente, cliente, canal).
   * Filtros, ordenações, salvamento de visões, compartilhamento, exportação (CSV/XLSX/PDF).
3. **KPIs padrão**:
   * Tickets abertos/fechados por período;
   * **TME** (Tempo Médio de Espera para primeira resposta);
   * **TMA** (Tempo Médio de Atendimento/chat);
   * **First Contact Resolution (FCR)**;
   * **SLA Met/Breakdown**;
   * **Produtividade por atendente** (tickets/dia, mensagens/dia);
   * **CSAT** pós-atendimento.
4. Agendamento de relatórios por e-mail.

### RF-11 – Automação e Bots (faseado)

1. **Boas-vindas** e triagem inicial no WhatsApp (menu rápido).
2. Respostas automáticas por horário (fora de expediente).
3. **Regras**: se {palavra-chave/cliente VIP} então {prioridade/roteamento/mensagem}.
4. Workflows visuais com gatilhos (webhook, status, tags) e ações (enviar msg, mover fila, setar campo, criar subtarefa).
5. Integração com **LLM** para sugestões de resposta (opcional).

### RF-12 – Portal do Cliente (opcional)

1. Acompanhar tickets, abrir novos, anexar arquivos.
2. Visualizar SLA, mensagens (resumo) e status.
3. Acesso à base de conhecimento pública.

### RF-13 – Auditoria e Logs

1. Registro de ações: criação/edição/transferência/fechamento, exclusões, exportações.
2. Trilha de mensagens e eventos de integração.
3. Relatórios de conformidade (LGPD).

### RF-14 – Administração e Configurações

1. Catálogos: categorias, prioridades, tags, motivos de encerramento.
2. Calendários e feriados.
3. Parâmetros de SLA e regras de escalonamento.
4. Gestão de sessões Evolution API (tokens, webhooks, rate limits).
5. Webhooks de saída (postbacks para ERP/CRM).
6. Multiempresa (marcas/unidades) com segregação lógica.

## 3. Requisitos Não Funcionais (RNF)

### RNF-01 – Arquitetura

* **Web** (SPA + API) com serviços desacoplados:
  + Front-end: React/Next.js ou Vue/Nuxt (i18n).
  + Back-end: Node.js/TypeScript (NestJS) ou .NET.
  + Mensageria: Kafka/RabbitMQ para eventos de mensagens e SLA.
  + Banco: **PostgreSQL** (transacional) + **Elastic/ClickHouse** (analytics, opcional).
  + Cache: Redis.
  + Armazenamento de mídia: S3 compatível.
  + Observabilidade: OpenTelemetry + Prometheus + Grafana.

### RNF-02 – Desempenho e Escalabilidade

* Suportar picos de **≥ 50 msg/seg** por instância de WhatsApp (ajustável).
* Tempo de resposta API P95 < **300 ms** (sem mídia).
* Processamento assíncrono de mídia e relatórios.
* Escalonamento horizontal de consumidores de mensagens.

### RNF-03 – Segurança

* TLS 1.2+ extremo a extremo.
* Criptografia at-rest (AES-256) para dados sensíveis.
* Secrets em cofre (KMS/Vault).
* RBAC e princípios de menor privilégio.
* Proteções anti-abuso (rate limiting, WAF).
* **LGPD**: bases legais, consentimento, minimização, retenção e anonimização/pseudonimização.

### RNF-04 – Confiabilidade

* Disponibilidade alvo **99,9%** mensal.
* Estratégia de **retry** e dead-letter para integrações.
* Backup diário + testes de restauração.
* Planos de contingência para quedas do provedor de WhatsApp (fila offline, retomada).

### RNF-05 – Usabilidade e Acessibilidade

* UI responsiva, WCAG 2.1 AA.
* Teclas de atalho no console do agente.
* Pesquisa global (tickets, clientes, mensagens).

### RNF-06 – Manutenibilidade

* Cobertura de testes 70%+ (unit/integration/e2e).
* Documentação OpenAPI/Swagger e Catálogo de Eventos.
* Versionamento semântico e migrações automatizadas.

## 4. Integração com WhatsApp via Evolution API

### 4.1 Conexão e Sessões

* Cadastro de **instâncias** (cada número/identidade).
* Campos: nome, número, webhook URL de entrada, token de autenticação, status (online/offline), data/hora de última sincronização.
* Healthcheck periódico; alerta em caso de desconexão.
* Suporte a **multi-sessão** por marca/unidade.

### 4.2 Webhooks (Incoming)

* Endpoint: POST /integrations/evolution/webhook/{instanceId}
* Eventos suportados (exemplos):
  + message.upsert (mensagem recebida)
  + message.status (status de entrega/leitura)
  + connection.update (mudanças de sessão)
  + presence.update
* **Validação**: assinatura HMAC + token de caminho + idempotency-key.
* **Mapeamento** para domínio interno (ticket/message).
* **Anti-loop**: ignorar eco de mensagens enviadas pelo próprio sistema (flag do origin).

### 4.3 Envio de Mensagens (Outgoing)

* Endpoint interno: POST /tickets/{id}/messages
* Payload: tipo (text, image, audio, document, location, contact), to, body, caption, mediaUrl/upload, context (reply), buttons (opcional).
* Adaptação para Evolution API (formato específico do provedor).
* Controle de **rate limit** por instância.
* Registro de **status** assíncrono via webhook message.status.

### 4.4 Tipos de Mensagens Suportados

* Texto, emoji, quebras de linha.
* Imagem (jpeg/png), Documento (pdf/doc/xls), Áudio (ogg/mp3/m4a), Vídeo (mp4\*), Localização, Contato.
* **Botões/menus rápidos** (quando suportados pela instância).
* Tamanho máximo e tipos permitidos configuráveis.

### 4.5 Riscos/Considerações

* Evolution API pode sofrer instabilidades por depender do cliente WhatsApp; prever **reconexão automática** e robusto mecanismo de **retry/backoff**.
* Respeitar políticas anti-spam e opt-in; bloqueio de disparos em massa.
* Monitorar mudanças de schema/versão do provedor.

## 5. Modelagem de Dados (Conceitual)

**Entidades Principais**  
- **User**(id, nome, email, perfil, timeId, status, mfaEnabled)  
- **Team**(id, nome, habilidades, horário, filaConfig)  
- **Client**(id, nome, documento?, segmento, slaPlanId, status)  
- **Contact**(id, clientId, nome, whatsapp, email, optOut, idioma)  
- **Ticket**(id, numero, clientId, contactId, canal, categoriaId, prioridade, status, slaId, agenteId?, createdAt, closedAt)  
- **TicketMessage**(id, ticketId, direction[in/out], type, body, mediaRef?, providerMsgId?, status, createdAt)  
- **SLA**(id, nome, priority, firstResponseTarget, resolutionTarget, calendarId, pauseStatuses)  
- **Category**(id, nome, parentId?)  
- **Tag**(id, nome, tipo) — relação N:N com Ticket  
- **Attachment**(id, ownerType, ownerId, filename, url, size, mime)  
- **KnowledgeArticle**(id, titulo, corpo, visibilidade, versao, autorId, status)  
- **AutomationRule**(id, nome, gatilho, condicoes, acoes, ativo)  
- **Survey**(id, tipo[CSAT/NPS], perguntas) e **SurveyResponse**(id, ticketId, score, comentario)  
- **EvolutionInstance**(id, nome, number, status, webhookUrl, token, rateLimit)  
- **AuditLog**(id, actorId, entidade, entidadeId, acao, diff, createdAt)

**Relacionamentos-chave**  
- Client 1—N Contact  
- Client 1—N Ticket  
- Ticket 1—N TicketMessage  
- Ticket N—N Tag  
- Team 1—N User; Team 1—N Ticket (fila)  
- SLA 1—N Ticket

## 6. Casos de Uso (principais)

### UC-01: Receber mensagem do WhatsApp e criar/associar Ticket

**Atores:** Evolution API (externo), Sistema, Atendente  
**Fluxo principal:**  
1. Webhook recebe message.upsert.  
2. Sistema valida assinatura e idempotência.  
3. Identifica contato por número; se inexistente, cria registro básico.  
4. Se não houver ticket aberto para o contato nos últimos X minutos (config), **cria** ticket; senão, **associa** ao ticket aberto.  
5. Notifica fila/time responsável.  
6. Exibe no console do agente.

**Extensões:**  
- 3a. Número em lista de bloqueio/opt-out ⇒ mensagem é registrada, mas **não** gera ticket automático (regra).  
- 4a. Política de “um ticket por assunto” via palavras-chave.

### UC-02: Responder cliente pelo console

**Atores:** Atendente  
**Fluxo:**  
1. Atendente abre ticket, digita resposta, anexa mídia, opcionalmente aplica template.  
2. Sistema envia via Evolution API, registra providerMsgId.  
3. Webhook atualiza message.status (entregue/lido).  
4. SLA de primeira resposta é cumprido.

### UC-03: Escalonamento por SLA

**Atores:** Motor de SLA, Gestor  
**Fluxo:**  
1. Job verifica SLAs ativos.  
2. Ticket prestes a romper ⇒ notifica atendente e gestor; aplica tag SLA-ALERTA.  
3. Em ruptura ⇒ transfere para fila de prioridade, notifica gerente e registra auditoria.

### UC-04: Encerramento e Pesquisa de Satisfação

**Atores:** Atendente, Cliente  
**Fluxo:**  
1. Atendente marca como **Resolvido**; sistema dispara CSAT via WhatsApp.  
2. Cliente responde (0–5).  
3. Sistema registra SurveyResponse e fecha ticket (ou reabre se resposta negativa e regra habilitada).

### UC-05: Indicadores Customizados

**Atores:** Gestor  
**Fluxo:**  
1. Gestor abre o construtor, escolhe dataset (**tickets**).  
2. Seleciona métricas (ex.: média de **primeira resposta** por **categoria** nos últimos 30 dias).  
3. Salva visão e agenda exportação semanal em PDF para e-mail.

## 7. Regras de Negócio (amostra)

* **RN-01:** Ticket é único por contato/assunto dentro de janela de 2h (configurável).
* **RN-02:** Status **Pendente Cliente** pausa SLA de resolução.
* **RN-03:** Mensagens proativas (sem ticket ativo) apenas para contatos com **opt-in**.
* **RN-04:** Anexos até 25 MB; formatos restritos por política.
* **RN-05:** Transferência de ticket não altera contagem de SLA.

## 8. UI/UX – Páginas e Componentes

* **Login/SSO**
* **Dashboard** (KPIs, filas, alertas de SLA)
* **Console do Agente**: lista de tickets (filtros por fila, status, SLA), painel de conversa, detalhes à direita (cliente, SLA, tags, atividades), templates/snippets, notas internas, macros.
* **Tickets**: grid com filtros salvos, views pessoais e compartilhadas.
* **Clientes & Contatos**: 360º, histórico, SLAs.
* **Base de Conhecimento**: editor, fluxo de publicação.
* **Relatórios**: construtor, biblioteca de visões, agendamentos.
* **Admin**: usuários, times, catálogos, SLA, integrações Evolution, webhooks, auditoria.

## 9. APIs Internas (exemplos em REST)

* POST /auth/login – autenticação.
* GET /tickets – listar (filtros: status, fila, cliente, data…).
* POST /tickets – criar ticket.
* GET /tickets/{id} / PATCH /tickets/{id}.
* POST /tickets/{id}/assign – atribuir/transferir.
* POST /tickets/{id}/messages – enviar mensagem.
* POST /integrations/evolution/webhook/{instanceId} – receber eventos.
* GET /reports/views / POST /reports/views – visões salvas.
* GET /analytics/kpi?name=SLA\_MET&from=... – KPIs.
* GET /admin/audit – auditoria.

**Autorização:** JWT com escopos; idempotência via Idempotency-Key em endpoints críticos.

## 10. Fluxos (alto nível)

### 10.1 Recebimento de Mensagem (ASCII)

Evolution API → Webhook /integrations/evolution/webhook/{instance}  
 ↳ Validação + Enfileira evento (Kafka/RabbitMQ)  
 ↳ Consumidor: Upsert Contact → Find/Upsert Ticket → Persist Message  
 ↳ Notificações (websocket) → Console do Agente

### 10.2 Envio de Mensagem

Agente no Console → POST /tickets/{id}/messages → Adapter Evolution  
 ↳ Evol. API ACK → Registra providerMsgId → Aguardando status  
 ↳ Webhook message.status → Atualiza status → Notifica Agente

### 10.3 SLA

Scheduler → Verifica clocks de SLA → Alerta/Escalonamento → Auditoria

## 11. Métricas e Indicadores (definições)

* **TME (Espera p/ 1ª resposta):** Δ entre 1ª msg recebida e 1ª msg do agente, desconsiderando horas fora de expediente.
* **TMA (Atendimento):** média de Δ entre msgs do agente durante janela ativa.
* **FCR:** % tickets resolvidos no primeiro contato (sem transferências/reaberturas).
* **SLA Met:** % tickets dentro dos alvos configurados.
* **Backlog:** tickets abertos por status/fila.
* **CSAT:** média/ distribuição de notas (0–5) na pesquisa de encerramento.

## 12. Plano de Testes (resumo)

### 12.1 Testes Funcionais

* **Integração WhatsApp**:
  + Recebimento de todos os tipos de mídia suportados;
  + Idempotência de mensagens;
  + Reconexão de sessão;
  + Status entregue/lido.
* **SLA**: pausa em Pendente Cliente; escalonamento correto; calendários.
* **Roteamento**: regras por skill e carga; transferência preserva histórico.
* **Construtor de indicadores**: filtros, agregações, exportação.

### 12.2 Testes Não Funcionais

* Carga: 50 msg/seg/instância com P95 < 300 ms.
* Resiliência: simular falhas do provedor (timeouts, HTTP 5xx).
* Segurança: testes de autorização, injeção, CSRF, rate limit.

### 12.3 Aceitação (UAT)

* Cenários ponta a ponta (UC-01 a UC-05).
* Critérios de aceite definidos por KPI mínimo (ex.: ≥ 90% SLA Met em sandbox controlado).

## 13. Critérios de Aceite (exemplos)

1. Sistema cria ticket automaticamente na primeira mensagem desconhecida e associa mensagens subsequentes ao mesmo ticket por 2h.
2. Console do agente exibe conversa em ≤ 2 s após recebimento.
3. Primeiro SLA configurado é respeitado e pausado em Pendente Cliente.
4. Gestor cria um indicador customizado e exporta CSV com os mesmos números do dashboard.
5. Pesquisa de CSAT é enviada e registrada no encerramento em ≥ 95% dos tickets fechados.

## 14. Segurança, Privacidade e LGPD

* **Bases Legais**: execução de contrato/legítimo interesse; consentimento para comunicações proativas.
* **Minimização**: armazenar apenas dados necessários (telefone, nome, histórico).
* **Retenção**: política por cliente (ex.: 24 meses), com anonimização/retenção seletiva de mídias.
* **Direitos do Titular**: export (portabilidade) e exclusão sob solicitação; registro no AuditLog.
* **Criptografia**: TLS em trânsito, at-rest para PII.
* **Segurança Operacional**: RBAC, logs imutáveis, segregação multiempresa.

## 15. Deployment e Operações

* **Ambientes**: dev, staging, prod.
* **CI/CD**: pipelines com testes automatizados, lint, SAST/DAST.
* **Infra**: Kubernetes + HPA; Ingress com TLS; storage S3; banco gerenciado.
* **Monitoramento**: healthchecks, SLOs, alertas (latência, erro 5xx, fila de mensagens, quedas de sessão Evolution).
* **Rotina de Backup/Restore** documentada e testada.

## 16. Roadmap (fases)

* **Fase 1 (MVP):** tickets WhatsApp, console agente, SLA básico, dashboards padrão, construtor simples de indicadores, CSAT, RBAC, auditoria.
* **Fase 2:** automações visuais, BK com sugestões, portal do cliente, NPS, templates avançados, multilíngue.
* **Fase 3:** IA assistiva (resumos, sugestões), previsão de ruptura de SLA, roteamento por ML, integrações ERP/CRM.

## 17. Riscos e Mitigações

* **Instabilidade do provedor WhatsApp/Evolution** → buffers, retries, fallback, comunicação de status aos agentes.
* **Bloqueios por uso indevido** → política de envios, opt-in/opt-out, limites e auditoria.
* **Escalada de custos de mídia/armazenamento** → políticas de retenção e expurgo automático.
* **Conformidade LGPD** → DPO envolvido, revisão periódica, relatórios.

## 18. Anexos (Futuros)

* Especificação OpenAPI.
* Catálogo de Eventos (JSON Schemas de webhooks).
* Protótipos de tela.
* Dicionário de Dados detalhado.

**FIM – v1.0**