**Documento de Gestão e Controle de Requisitos para Desenvolvimento de Software**

- Data de Criação: 13 de setembro de 2025

- Autor: Equipe de desenvolvimento e gestão de requisitos.

- Aprovação: Pendente (revisão por stakeholders)

- Histórico de Mudanças:

|  |  |  |  |
| --- | --- | --- | --- |
| Versão | Data | Descrição das Mudanças | Autor |
| 1.0 | 13/09/25 | Criação de documento unificado para gestão e controle de requisitos. | Julio César Morais |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Este documento segue as boas práticas recomendadas, inspiradas no padrão IEEE 830 para Especificação de Requisitos de Software (SRS), adaptadas para um contexto ágil. Ele serve como artefato vivo para a gestão e controle de requisitos, garantindo rastreabilidade, validação e priorização. O foco é no desenvolvimento de uma Plataforma de Delivery de Alimentos (semelhante ao iFood), com ênfase em requisitos funcionais e não funcionais de alto nível.

Para colaboração, recomenda-se o uso de ferramentas como Confluence, Notion ou Google Docs. A validação deve envolver equipe de desenvolvimento, gerentes de projeto e stakeholders (clientes, restaurantes, entregadores).

---

**1. Introdução**

**1.1 Objetivo**

Este documento estabelece uma base clara e compartilhável para os requisitos do software, descrevendo o que o sistema deve fazer (funcionais) e como deve se comportar (não funcionais). Ele destina-se a:

- Equipe de desenvolvimento (para implementação e testes).

- Gerentes de projeto (para planejamento e rastreabilidade).

- Stakeholders (clientes, restaurantes, entregadores) para alinhamento e validação.

- Evitar ambiguidades e scope creep, promovendo um desenvolvimento eficiente.

**1.2 Escopo do Produto**

O que o software vai fazer:

- Permitir cadastro, pedidos, entregas e pagamentos para clientes, restaurantes e entregadores.

- Integrar geolocalização, pagamentos e notificações em uma plataforma web.

Posteriormente integrado para ambiente mobile

O que o software **NÃO** vai fazer (fora do escopo inicial):

- Gerenciar estoques físicos de restaurantes (apenas cardápios digitais).

- Suportar entregas internacionais ou moedas além de BRL.

- Incluir funcionalidades de e-commerce para produtos não alimentícios.

**1.3 Definições, Acrônimos e Abreviações**

| Termo/Acrônimo | Definição

|  |  |
| --- | --- |
| RF | Requisito Funcional: Descreve operações e fluxos de negócio. |
| RNF | Requisito Não Funcional: Descreve qualidades como performance e segurança. |
| OTP | One-Time Password: Senha de uso único para verificação. |
| LGPD | Lei Geral de Proteção de Dados (Brasil). |
| GDPR | General Data Protection Regulation (Europa). |
| WCAG | Web Content Accessibility Guidelines. |
| SRS | Software Requirements Specification. |
| MoSCoW | Método de priorização: Must (essencial), Should (importante), Could (desejável), Won't (fora do escopo). |
|  |  |

**1.4 Referências**

- Padrão IEEE 830: Guia para Especificação de Requisitos de Software.

- Artigo: "Utilização de Checklist para Validação de Requisitos de Software" (https://imasters.com.br/devsecops/utilizacao-de-checklist-para-validacao-de-requisitos-de-software).

- Protótipos iniciais e estudos de viabilidade (a serem anexados em versões futuras)

**2. Descrição Geral**

**2.1 Perspectiva do Produto**

A plataforma é um novo sistema de delivery de alimentos, integrado a ecossistemas existentes como APIs de geolocalização (Google Maps) e pagamentos (Stripe/PagSeguro). Ela visa otimizar fluxos de pedidos em um contexto urbano brasileiro, competindo com apps como iFood, com foco em usabilidade e escalabilidade.

**2.2 Características do Usuário**

Os usuários são divididos em papéis principais:

- **Cliente**: Usuário final que faz pedidos; perfil típico: jovem adulto, urbano, com smartphone.

- **Restaurante**: Proprietário/gerente que gerencia cardápios e pedidos; precisa de dashboards analíticos.

- **Entregador**: Freelancer com veículo; requer rastreamento em tempo real e status de disponibilidade.

- **Administrador**: Equipe interna para aprovações e suporte.

**2.3 Restrições**

- Tecnológicas: Desenvolvimento em linguagens modernas (ex.: React Native para mobile, Node.js para backend); sem suporte a browsers legados.

- Segurança: Conformidade obrigatória com LGPD/GDPR; criptografia AES-256 para dados sensíveis.

- Ambiente: Compatibilidade com Android 8.0+ e iOS 12+; operação em redes de baixa largura de banda (suporte offline parcial).

**3. Requisitos Funcionais (RF)**

Os requisitos funcionais definem as operações principais. Cada um é identificado, descrito e priorizado via MoSCoW. Eles foram derivados da lista inicial fornecida, refinados para clareza e rastreabilidade. Formato inspirado em User Stories para agilidade.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Nome | Descrição (User Story) | Critério de Aceitação |
| RF-01 | Cadastro e Login de Cliente | Como um novo cliente, eu quero me cadastrar/login via e-mail, telefone ou redes sociais para acessar funcionalidades, com verificação por OTP. | Confirmação por OTP/email. |
| RF-02 | Cadastro e Login de Restaurante | Como um restaurante, eu quero me cadastrar/login com aprovação administrativa para gerenciar perfil (fotos, horários, cardápio). | Edição de perfil. |
| RF-03 | Cadastro e Login de Entregador | Como um entregador, eu quero me cadastrar/login com verificação de documentos para definir status online/offline. | Histórico de verificações. |
| RF-04 | Busca de Restaurantes | O sistema deve exibir lista de restaurantes próximos, filtrável por localização, tipo de cozinha, avaliações, preço e tempo estimado. | Ordenação por relevância. |
| RF-05 | Visualização de Cardápio | O sistema deve permitir navegação em cardápios com detalhes (fotos, descrições, customizações, promoções). | Opções de alergênicos. |
| RF-06 | Carrinho de Compras | O sistema deve adicionar itens ao carrinho, calculando subtotal, taxas e descontos. | Cálculo automático. |
| RF-07 | Processamento de Pedidos | O sistema deve processar pedidos com seleção de endereço, pagamento e confirmação por push. | Notificação imediata. |
| RF-08 | Notificação a Restaurantes | O sistema deve notificar restaurantes sobre pedidos, permitindo aceitação/rejeição e atualização de status. | Status: Preparando, Pronto, Em Rota. |
| RF-09 | Alocação de Entregadores | O sistema deve alocar entregadores próximos via geolocalização e notificar atribuição. | Aceitação em 2min. |
| RF-10 | Rastreamento em Tempo Real | O sistema deve fornecer mapa de rastreamento para cliente e restaurante. | ETA dinâmico. |
| RF-11 | Processamento de Pagamentos | O sistema deve processar pagamentos pós-entrega, com reembolso em cancelamentos. | Histórico de transações. |
| RF-12 | Avaliações e Reviews | O sistema deve permitir avaliações pós-pedido (estrelas, comentários, fotos). | Moderação de conteúdo. |
| RF-13 | Dashboards Analíticos | O sistema deve fornecer relatórios para restaurantes (vendas) e entregadores (ganhos). | Exportação CSV. |
| RF-14 | Suporte ao Cliente | O sistema deve suportar chat/tickets para resolução de problemas. | Integração com IA para respostas rápidas. |
| RF-15 | Gerenciamento de Promoções | O sistema deve gerenciar cupons com regras automáticas (ex.: frete grátis >R$50 | Relatórios de uso. |
|  |  |  |  |

Controle: Cada RF será rastreado por ID de tarefas criadas dentro da plataforma do GitHub para gestão de atividades. Validação via checklist (ver Apêndice A).

**4. Requisitos Não Funcionais (RNF)**

Os RNF definem qualidades mensuráveis. Categorias baseadas em práticas padrão, com métricas concretas.

| ID | Categoria | Descrição | Métrica/Testável | Prioridade (MoSCoW) |

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Categoria | Descrição | Critério de aceitação |
| RNF-01 | Performance | Responder a buscas de restaurantes em <2s, com 10.000 usuários simultâneos. | | Tempo de resposta <2s (95% casos). |
| RNF-02 | Escalabilidade | Suportar escalonamento para 1M usuários mensais via microserviços. | Throughput: 1.000 req/s. |
| RNF-03 | Disponibilidade | Manter uptime de 99,9% com redundância geográfica. | Downtime <8,76h/ano. |
| RNF-04 | Segurança | Criptografar dados sensíveis (AES-256 em repouso, TLS 1.3 em trânsito); conformidade LGPD/GDPR | Auditoria anual. |
| RNF-05 | Autenticação | Implementar 2FA obrigatória para admins, opcional para usuários. | Taxa de adesão >80%. |
| RNF-06 | Usabilidade | Interface intuitiva, responsiva (mobile/web), WCAG 2.1 AA. | Tempo de tarefa <3 cliques. |
| RNF-07 | Portabilidade | Compatível com Android 8.0+/iOS 12+; suporte offline para cardápios. | Testes em 5 dispositivos. |
| RNF-08 | Manutenibilidade | Seguir SOLID/Clean Architecture; cobertura de testes >80%. | Métricas SonarQube. |
| RNF-09 | Integração | Integrar APIs externas (Google Maps, pagamentos) com retry em falhas. | Taxa de sucesso >99%. |
| RNF-10 | Log e Monitoramento | Registrar logs de auditoria; alertas em tempo real para fraudes (ML). | Logs retidos 90 dias. |
| RNF-11 | Internacionalização | formatação local. | Testes linguísticos. |
| RNF-12 | Sustentabilidade | Otimizar energia com caching (Redis), reduzindo latência em 50%. | Consumo CPU <70% em pico. |

Controle: RNFs serão validados via testes automatizados (ex.: load testing com JMeter). Rastreabilidade ligada a RFs relevantes.

**5. Gestão e Controle de Requisitos**

**5.1 Priorização e Refinamento**

- Método: MoSCoW para priorização inicial; refinamento em sprints ágeis via User Stories e Casos de Uso.

- Rastreabilidade: Matriz de rastreabilidade (a ser anexada) mapeia RF/RNF a user stories, testes e código.

- Mudanças: Qualquer alteração requer aprovação via Change Request Form, com impacto analysis (custo, risco).

**5.2 Validação e Verificação**

- Checklist de Validação (baseado no artigo referenciado):

1. Requisito é claro e sem ambiguidades? (Sim/Não).

2. É mensurável/testável? (Critérios definidos).

3. Alinha-se ao escopo? (Sim/Não).

4. Priorizado adequadamente? (MoSCoW).

5. Rastreado a stakeholders? (Sim/Não).

- Revisões: Semanais com equipe; validação final com stakeholders antes de cada sprint.

**5.3 Versionamento e Colaboração**

- Ferramentas: Notion para edição colaborativa; Git para versionamento de artefatos relacionados.

- Auditoria: Logs de mudanças no histórico acima.

**Apêndice A: Checklist para Validação de Requisitos**

(Adaptado do artigo iMasters)

- [ ] Requisitos funcionais cobrem fluxos end-to-end?

- [ ] Não funcionais são mensuráveis?

- [ ] Todos os papéis de usuário estão representados?

- [ ] Riscos identificados (ex.: dependências externas)?

Este documento pode ser expandido com diagramas UML (ex.: casos de uso) ou priorização detalhada. Para discutir arquitetura técnica, como microsserviços ou integrações.