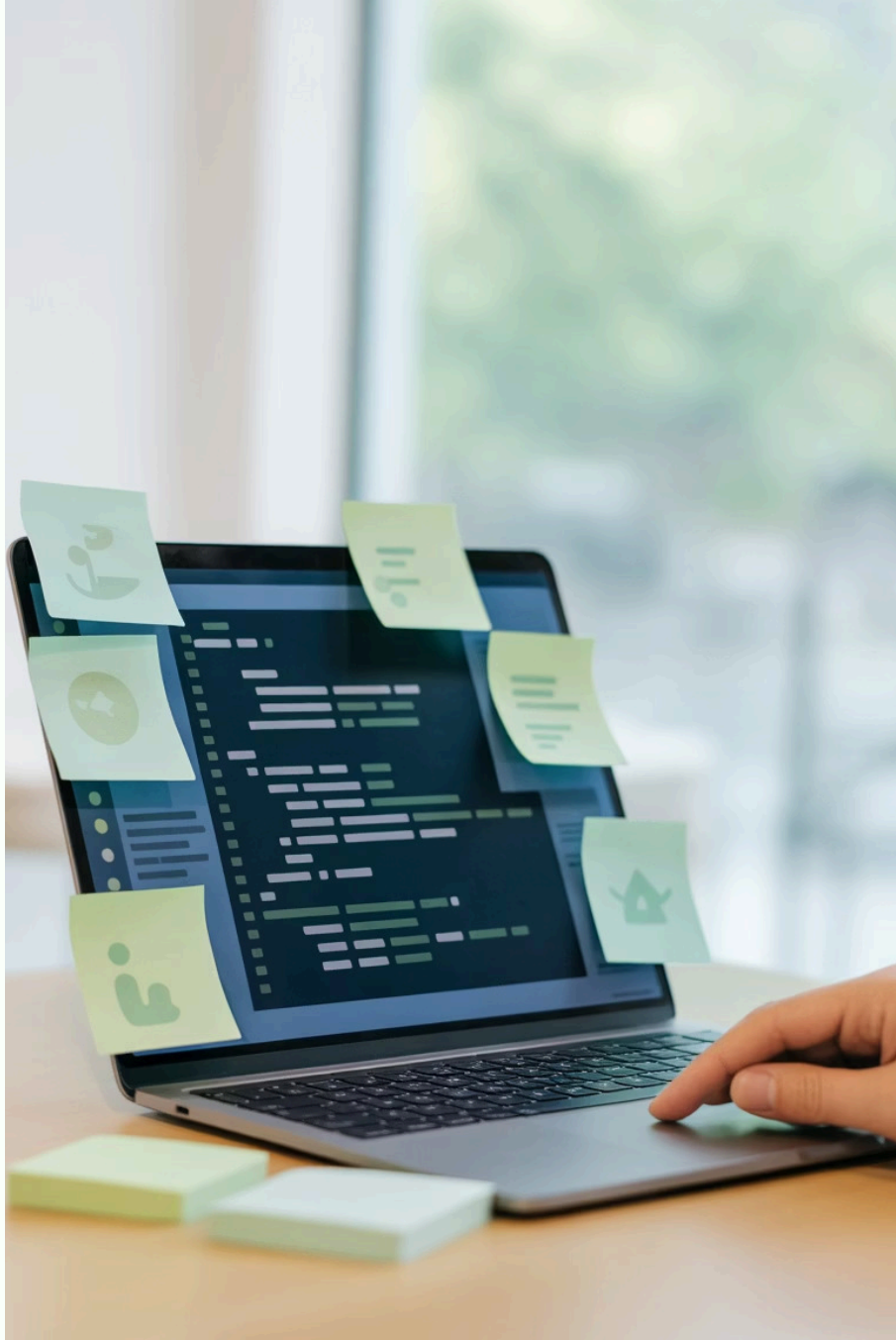
Gestão de Requisitos e Notação Markdown

Engenharia de Software II

Análise e Desenvolvimento de Sistemas



Agenda da Aula

1

Fundamentos de Gestão de Requisitos

Conceitos básicos, falhas comuns e papéis dos requisitos

2

Tipos de Requisitos

Funcionais, não funcionais e restrições técnicas

3

Critérios de Requisitos

Aceitação, aceitabilidade, qualidade, rastreabilidade, priorização e testabilidade

4

Documentação de Requisitos

Padronização, clareza e boas práticas

5

Markdown para Requisitos

Sintaxe, templates e aplicação prática

6

Oficina Prática

Exercícios de aplicação dos conceitos aprendidos

Fundamentos de Gestão de Requisitos

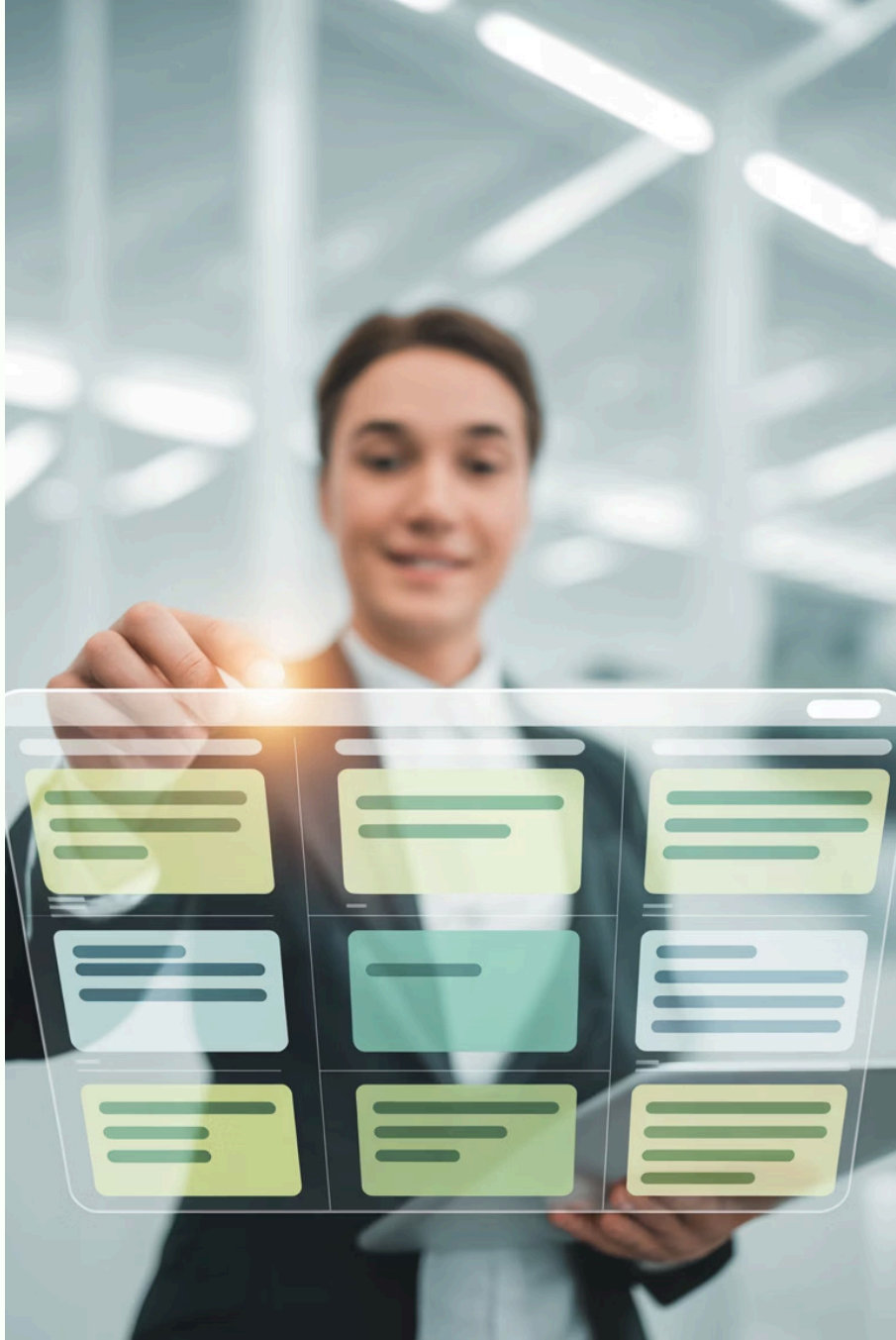
Bloco 1

Base para o desenvolvimento de software de qualidade

O que é Gestão de Requisitos?

É o processo sistemático de **documentar**, **analisar**, **priorizar**, **rastrear** e **validar** as necessidades dos clientes e stakeholders ao longo de todo o ciclo de vida do software.

A gestão de requisitos não é um evento pontual, mas um processo contínuo que acompanha o desenvolvimento desde a concepção até a manutenção do produto.



Por que requisitos falham em projetos?

Requisitos incompletos

Faltam detalhes críticos que só são percebidos durante o desenvolvimento

Comunicação falha

Mal-entendidos entre stakeholders, desenvolvedores e testadores

Falta de priorização

Equipe gasta tempo em funcionalidades menos relevantes

Mudanças sem controle

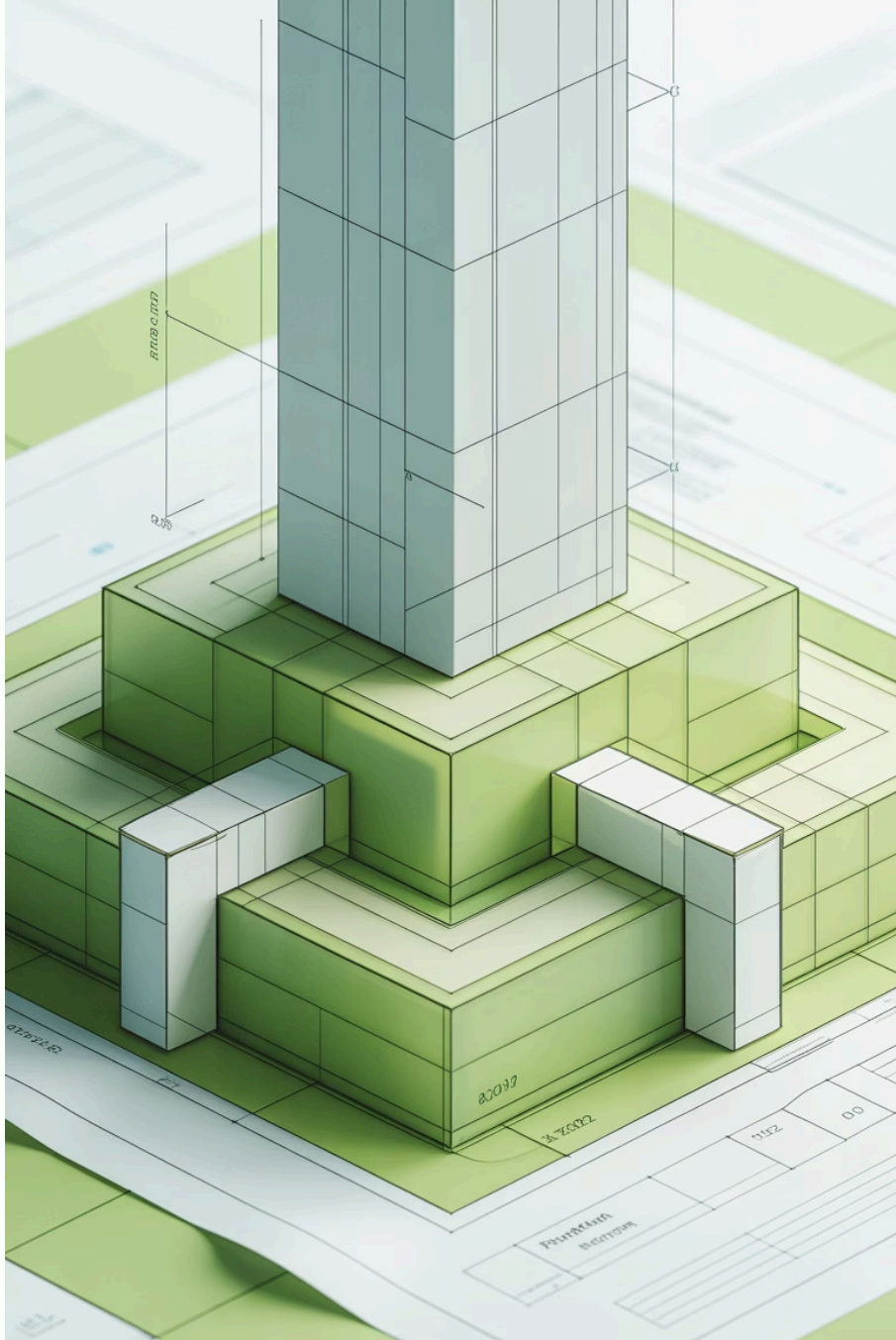
Alterações constantes sem documentação ou impacto avaliado

Segundo o PMI, cerca de 47% dos projetos falham por problemas relacionados a requisitos mal definidos.

Papel dos requisitos no ciclo de vida

Os requisitos estabelecem a fundação para todas as fases subsequentes do desenvolvimento:

- Servem como **base para o design** da arquitetura e interfaces
- Orientam as **decisões de desenvolvimento** e implementação
- Fornecem os **critérios para testes** e validação
- Guiam as **atividades de manutenção** e evolução
- Estabelecem **métricas de sucesso** para o projeto



Requisitos e qualidade do sistema

Consequências de requisitos bem definidos:

- Redução de retrabalho
- Aumento da confiabilidade
- Melhor estimativa de prazos
- Redução de custos
- Maior satisfação dos usuários

"A qualidade do software é diretamente proporcional à qualidade dos seus requisitos." – Karl Wiegers

Stakeholders e requisitos

Usuários finais

Experiência de uso, funcionalidades esperadas

Jurídico

Conformidade legal, LGPD, normas

Clientes

Valor de negócio, orçamento, prazos

Desenvolvedores

Viabilidade técnica, manutenibilidade

QA

Testabilidade, critérios de aceitação

Cada stakeholder traz diferentes necessidades, restrições e expectativas que precisam ser equilibradas nos requisitos.

Tipos de Requisitos

Bloco 2

Diferentes categorias para diferentes necessidades

Requisitos Funcionais

Requisitos Funcionais definem o que o sistema deve fazer:

- Descrevem **funções** que o sistema deve executar
- Especificam **comportamentos** esperados em determinadas condições
- Detalham **serviços** que o sistema deve oferecer
- Estabelecem **interações** entre o sistema e seu ambiente
- Normalmente são expressos na forma "**O sistema deve...**"

Exemplos de Requisitos Funcionais

RF001

O sistema deve permitir que usuários realizem login utilizando email e senha.

RF002

O sistema deve enviar um email de confirmação após o cadastro de um novo usuário.

RF003

O sistema deve permitir a filtragem de produtos por categoria, preço e avaliação.

Requisitos funcionais são específicos e detalhados, descrevendo exatamente o que o sistema deve fazer, sem ambiguidades.

Requisitos Não Funcionais

Requisitos Não Funcionais definem como o sistema deve ser ou quais qualidades deve possuir:

- Desempenho (tempo de resposta, throughput)
- Segurança (autenticação, autorização)
- Usabilidade (facilidade de uso, acessibilidade)
- Confiabilidade (disponibilidade, tolerância a falhas)
- Escalabilidade (capacidade de crescimento)
- Manutenibilidade (facilidade de correção e evolução)

Exemplos de Requisitos Não Funcionais

RNF001 - Desempenho

O tempo de resposta para qualquer operação no sistema deve ser inferior a 2 segundos para 95% das requisições sob carga normal.

RNF002 - Segurança

Todas as senhas armazenadas no sistema devem ser criptografadas utilizando algoritmo bcrypt com salt de pelo menos 10 rounds.

RNF003 - Usabilidade

O sistema deve ser acessível para usuários com deficiência visual, seguindo as diretrizes WCAG 2.1 nível AA.

Restrições e requisitos técnicos

Restrições são limitações que afetam como o sistema deve ser desenvolvido:

- **Restrições de ambiente:** "Deve funcionar em ambiente AWS"
- **Restrições de plataforma:** "Deve ser compatível com iOS 14 ou superior"
- **Restrições de linguagem:** "Deve ser desenvolvido em Java 11"
- **Restrições normativas:** "Deve atender à ISO 27001"
- **Restrições de prazo:** "Deve ser entregue em 3 meses"
- **Restrições de orçamento:** "Custo de infraestrutura não deve exceder R$5.000/mês"

Critérios de Requisitos

Bloco 3

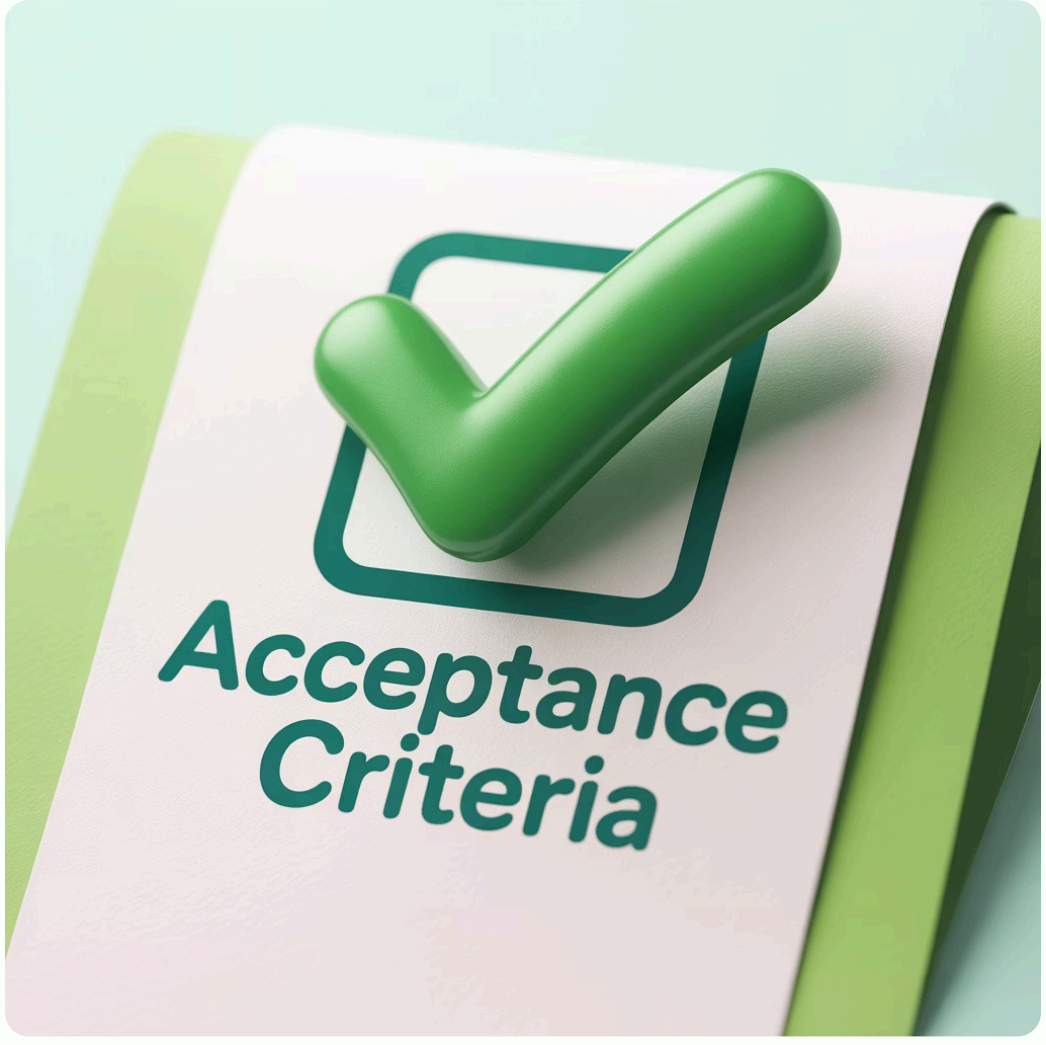
Garantindo requisitos claros, testáveis e alinhados às necessidades

Critérios de Aceitação

Condições objetivas e específicas que comprovam se um requisito foi atendido corretamente.

Características ideais:

- Objetivos e mensuráveis
- Verificáveis por testes
- Binários (atendido ou não atendido)
- Compreensíveis para todos os envolvidos
- Alinhados ao valor de negócio



Exemplo de Critérios de Aceitação

RF001: Cadastro de Produto

O sistema deve permitir que usuários com perfil de administrador cadastrem novos produtos.

Critérios de Aceitação:

1. Após preencher todos os campos obrigatórios e clicar em "Salvar", o produto deve aparecer na listagem imediatamente.
2. Caso algum campo obrigatório não seja preenchido, o sistema deve exibir mensagem de erro indicando os campos faltantes.
3. Ao cadastrar um produto com código já existente, o sistema deve exibir mensagem de erro informando sobre a duplicidade.
4. Apenas usuários com perfil "Administrador" ou "Gerente de Produtos" devem visualizar a opção de cadastro.

Critérios de Aceitabilidade

Conformidade com o negócio

O requisito atende a uma necessidade real do negócio e está alinhado à estratégia da empresa?

Conformidade com o usuário

O requisito atende às expectativas e necessidades dos usuários finais?

Conformidade legal

O requisito atende às legislações aplicáveis (LGPD, Marco Civil, acessibilidade, etc.)?

Viabilidade técnica

O requisito é tecnicamente viável dentro das restrições do projeto?

Exemplo de Critérios de Aceitabilidade

RNF004: Proteção de Dados Pessoais

O sistema deve implementar medidas de proteção para dados pessoais dos usuários.

Critérios de Aceitabilidade:

- **Conformidade legal:** Implementação seguindo a LGPD (Lei Geral de Proteção de Dados)
- **Segurança:** Dados pessoais devem ser criptografados em repouso e em trânsito
- **Transparência:** Usuários devem poder visualizar, corrigir e excluir seus dados
- **Consentimento:** Coleta de dados deve ter consentimento explícito do usuário

Critérios de Qualidade

Definem características de excelência nos requisitos não funcionais:

- **Desempenho:** tempos de resposta, capacidade, latência
- **Escalabilidade:** capacidade de crescimento
- **Segurança:** proteção contra ameaças
- **Usabilidade:** facilidade de uso e aprendizado
- **Confiabilidade:** disponibilidade, tolerância a falhas

Critérios de Rastreabilidade

Garantem que cada requisito possa ser rastreado desde sua origem até sua implementação e validação.

Uma boa rastreabilidade estabelece:

- De onde veio o requisito (stakeholder, documento, lei)
- Onde está implementado (módulos, classes)
- Como é testado (casos de teste)
- Dependências entre requisitos (pré-requisitos)
- Histórico de alterações (versionamento)

Critérios de Priorização

MoSCoW

Must Have, Should Have, Could Have, Won't Have

Modelo Kano

Obrigatórios, Desempenho, Atratividade, Indiferentes

WSJF

Weighted Shortest Job First (valor ÷ tamanho)

A priorização adequada garante que o time foque primeiro no que gera mais valor para o cliente e o negócio.

Critérios de Testabilidade

Asseguram que o requisito possa ser verificado objetivamente.

Um requisito é testável quando:

- É **observável** (comportamento visível)
- É **específico** (sem ambiguidades)
- É **mensurável** (podemos determinar se foi atingido)
- É **independente** ou tem dependências claras
- Possui **critérios de aceitação** bem definidos

Documentação de Requisitos

Bloco 4

A clareza na comunicação das necessidades

Importância da padronização

Padronizar a documentação de requisitos traz diversos benefícios:

- **Reduz ambiguidades** e interpretações divergentes
- **Facilita a comunicação** entre membros da equipe
- **Melhora a rastreabilidade** de requisitos ao longo do projeto
- **Simplifica a validação** dos requisitos pelos stakeholders
- **Aumenta a produtividade** no desenvolvimento e testes
- **Garante consistência** nas entregas e documentação

Ambiguidade em requisitos

Requisitos ambíguos:

⊗ "O sistema deve ser rápido."

"A interface deve ser amigável."

"O usuário pode cancelar a operação."

"O sistema deve ter boa performance."

Problemas causados:

- Diferentes interpretações pela equipe
- Impossibilidade de validação objetiva
- Discussões e retrabalho
- Desalinhamento entre expectativa e entrega

Requisitos ambíguos são uma das principais causas de falha em projetos de software.

Requisitos vagos x testáveis

Vago

"O sistema deve ser rápido."

Testável

"O sistema deve responder a qualquer consulta em até 2 segundos para 95% das requisições."

Vago

"O sistema deve ser fácil de usar."

Testável

"Usuários sem treinamento prévio devem conseguir completar o cadastro em menos de 5 minutos."

Vago

"O sistema deve suportar muitos usuários."

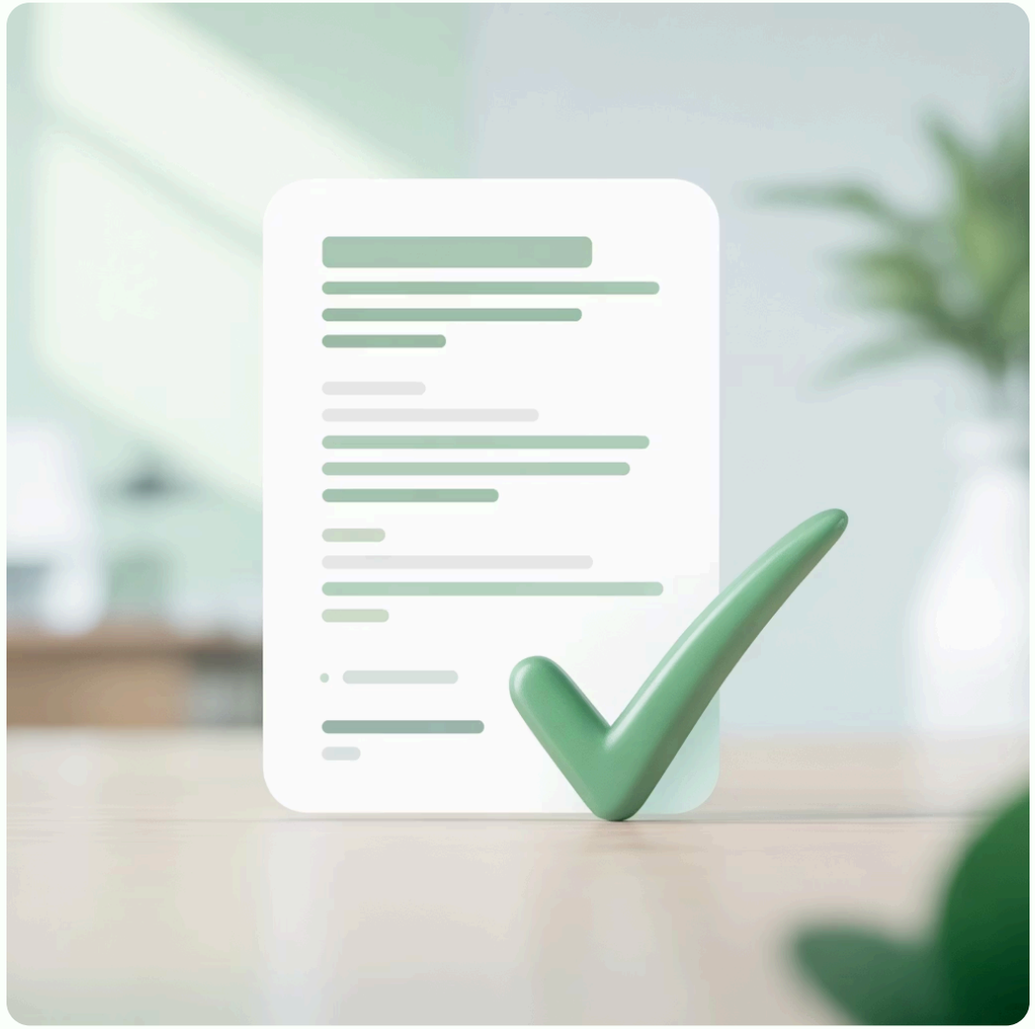
Testável

"O sistema deve suportar 1.000 usuários simultâneos com degradação máxima de 10% no tempo de resposta."

Linguagem simples e objetiva

Recomendações:

- Use frases curtas e diretas
- Evite termos técnicos excessivos
- Seja preciso nos valores e métricas
- Use verbos na voz ativa
- Defina termos específicos do domínio
- Evite palavras subjetivas (bom, melhor, etc.)



A linguagem simples facilita o entendimento por todos os stakeholders, independente de seu background técnico.

Rastreabilidade como documento vivo

Requisitos evoluem junto com o projeto. É essencial:

- Manter o histórico de alterações dos requisitos
- Documentar justificativas para mudanças
- Atualizar dependências entre requisitos
- Comunicar alterações a todos os envolvidos
- Revisar impactos em escopo, prazo e custo
- Garantir aprovação formal para mudanças significativas

Um sistema de controle de versão como o Git é essencial para gerenciar esse histórico.

Más práticas na documentação

Requisitos duplicados

Geram confusão e aumentam o risco de inconsistências quando apenas uma das cópias é atualizada.

Requisitos inconsistentes

Contradições entre requisitos que tornam impossível atender a ambos simultaneamente.

Requisitos sem fonte clara

Impossibilita validar se ainda são relevantes ou esclarecer dúvidas sobre seu propósito.

Requisitos excessivamente técnicos

Difíceis de entender pelos stakeholders não técnicos, prejudicando a validação.

Introdução ao Markdown

Bloco 5

Uma linguagem simples para documentação poderosa

O que é Markdown?

Markdown é uma linguagem leve de marcação criada por John Gruber em 2004, que permite formatar texto utilizando uma sintaxe simples e intuitiva.

Principais características:

- Sintaxe simples e legível
- Facilmente convertível para HTML
- Não requer ferramentas especializadas
- Foco no conteúdo, não na formatação
- Amplamente suportado em plataformas de desenvolvimento

Markdown em Engenharia de Software

Documentação de Requisitos

Estruturação clara com formatação simples para requisitos e critérios

Manuais Técnicos

Documentação de APIs, instalação e configuração de sistemas

Checklists de Teste

Criação de casos de teste com passos estruturados

Documentação de Código

Explicações sobre componentes e arquitetura

O Markdown é especialmente útil para documentação técnica devido à facilidade de incluir blocos de código.

Sintaxe: Títulos

Markdown:

```
# Título de Nível 1
## Título de Nível 2
### Título de Nível 3
#### Título de Nível 4
```

Resultado:

Título de Nível 1

Título de Nível 2

Título de Nível 3

Título de Nível 4

Títulos são criados usando o símbolo `#` seguido de espaço. O número de `#` indica o nível do título.

Sintaxe: Listas

Markdown:

```
- Item 1
- Item 2
  - Subitem 2.1
  - Subitem 2.2


1. Primeiro item
2. Segundo item
3. Terceiro item
```

Resultado:

Lista não ordenada:

- Item 1
- Item 2
  - Subitem 2.1
  - Subitem 2.2

Lista ordenada:

1. Primeiro item
2. Segundo item
3. Terceiro item

Sintaxe: Negrito e Itálico

Markdown:

```
**Texto em negrito**
__Também em negrito__

*Texto em itálico*
_Também em itálico_

***Negrito e itálico***
```

Resultado:

Texto em negrito
Também em negrito

Texto em itálico
Também em itálico

Negrito e itálico

A formatação com `*` ou `_` é essencial para destacar informações importantes nos requisitos.

Sintaxe: Tabelas

Markdown:

```
| Coluna A | Coluna B | Coluna C |
|----------|----------|----------|
| Valor A1 | Valor B1 | Valor C1 |
| Valor A2 | Valor B2 | Valor C2 |
| Valor A3 | Valor B3 | Valor C3 |
```

Resultado:

Coluna A	Coluna B	Coluna C
Valor A1	Valor B1	Valor C1
Valor A2	Valor B2	Valor C2
Valor A3	Valor B3	Valor C3

Tabelas são úteis para comparar requisitos, mostrar matrizes de rastreabilidade ou organizar critérios.

Sintaxe: Blocos de Código

Markdown:

````
```
function exemplo() {
  console.log("Olá, mundo!");
}
```

Código inline: `var x = 10;`
````

Resultado:

```
function exemplo() {
  console.log("Olá, mundo!");
}
```

Código inline: `var x = 10;`

Blocos de código são essenciais para documentar exemplos técnicos, como APIs, configurações ou protótipos.

Markdown em ferramentas

O Markdown é amplamente suportado em ferramentas modernas de desenvolvimento:

- **GitHub:** README, issues, pull requests, wikis
- **GitLab:** documentação de projetos e discussões
- **Trello:** descrições de cards
- **Slack:** mensagens e snippets
- **Notion:** documentação e wikis
- **Jira:** descrição de tarefas e comentários
- **VS Code:** visualização de arquivos .md

A ampla adoção torna o Markdown uma habilidade fundamental para profissionais de desenvolvimento.

Template de Requisitos em Markdown

Bloco 6

Padronizando a documentação para maior clareza

Estrutura de requisito

Um template básico de requisito em Markdown inclui:

```
# [ID] - [Título do Requisito]

## Descrição
[Descrição detalhada do requisito]

## Critérios Associados
- [Lista de critérios aplicáveis]

## Histórico de Alterações
| Data | Responsável | Alteração |
|------|-------------|-----------|
| dd/mm/aaaa | Nome | Descrição da alteração |
```

Seção: Critérios de Aceitação

Template:

```
## Critérios de Aceitação

1. [Critério 1]
2. [Critério 2]
3. [Critério 3]
...
```

Exemplo:

```
## Critérios de Aceitação

1. Após submeter o formulário com
dados válidos, o usuário deve
receber confirmação visual.

2. Se algum campo obrigatório não
for preenchido, o sistema deve
exibir mensagem de erro específica.

3. Ao cadastrar CPF já existente,
o sistema deve alertar o usuário.
```

Use listas numeradas para facilitar a referência a critérios específicos durante testes e validações.

Seção: Critérios de Aceitabilidade

```
## Critérios de Aceitabilidade

### Conformidade Legal
- [Requisitos legais, regulatórios ou de compliance]

### Usabilidade
- [Requisitos de experiência do usuário]

### Regras de Negócio
- [Alinhamento com processos e políticas de negócio]

### Viabilidade Técnica
- [Considerações sobre implementação e tecnologia]
```

Esta seção garante que o requisito esteja alinhado ao contexto mais amplo do negócio e suas restrições.

Seção: Critérios de Qualidade

Critérios de Qualidade

Performance
- [Tempo de resposta, throughput, capacidade]

Segurança
- [Proteção de dados, autenticação, autorização]

Confiabilidade
- [Disponibilidade, tolerância a falhas, recuperação]

Manutenibilidade
- [Facilidade de correção, adaptação, extensão]

Seção: Critérios de Rastreabilidade

Template:

```
## Rastreabilidade

### Origem
- [Fonte do requisito]

### Casos de Teste
- [IDs dos casos de teste]

### Dependências
- [Pré-requisitos]
- [Requisitos relacionados]
```

Exemplo:

```
## Rastreabilidade

### Origem
- Entrevista com Gerente de Vendas
- Ata de Reunião #28 (10/05/2023)

### Casos de Teste
- TC-001, TC-002, TC-045

### Dependências
- RF-015: Autenticação de Usuário
- RF-022: Cadastro de Produtos
```

Seção: Critérios de Priorização

MoSCoW

```
## Priorização
**Prioridade:** Must Have

*Justificativa: Funcionalidade
essencial para o MVP, sem a
qual o sistema não atende o
objetivo principal.*
```

Escala Numérica

```
## Priorização
**Prioridade:** Alta (5/5)

*Justificativa: Impacto crítico
na experiência do usuário e
no valor de negócio.*
```

A priorização clara ajuda a equipe a focar no que realmente importa e gerenciar expectativas dos stakeholders.

Seção: Critérios de Testabilidade

```
## Testabilidade

### Abordagem de Teste
- [Estratégia para testar o requisito: manual, automatizado, etc.]

### Pré-condições
- [Estado inicial necessário para teste]

### Dados de Teste
- [Dados necessários para executar os testes]

### Cobertura Esperada
- [Cenários que devem ser cobertos pelos testes]
```

Esta seção é especialmente útil para a equipe de QA, facilitando o planejamento e execução dos testes.

Exemplo completo

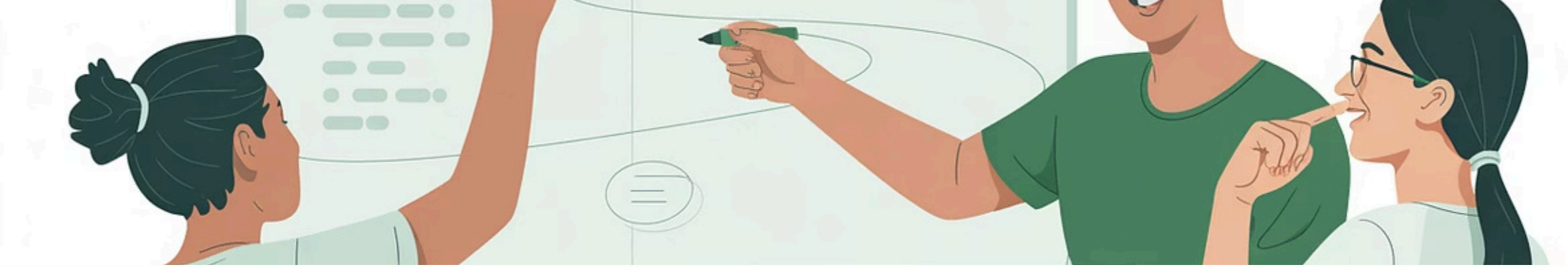
RF-001: Cadastro de Produto

```
# RF-001: Cadastro de Produto

## Descrição
O sistema deve permitir que usuários com perfil de administrador
cadastrem novos produtos no catálogo da loja.

## Critérios de Aceitação
1. Após preencher todos os campos obrigatórios e clicar em "Salvar",
 o produto deve aparecer na listagem imediatamente.
2. Caso algum campo obrigatório não seja preenchido, o sistema deve
 exibir mensagem de erro.
3. Ao cadastrar um produto com código já existente, o sistema deve
 exibir mensagem de erro.

## Critérios de Qualidade
- Tempo de resposta: submissão do formulário em até 1 segundo
- Validação de campos em tempo real (client-side)

## Priorização
**Prioridade:** Must Have

## Rastreabilidade
- **Origem:** Reunião de Requisitos #3 (15/04/2023)
- **Dependências:** RF-015 (Autenticação de Usuário)
```

Oficina Prática

Bloco 7

Aplicando os conceitos na prática



Atividade 1

Crie um requisito funcional simples em Markdown

Em grupos de 2-3 pessoas, criem um requisito funcional seguindo o template apresentado:

1. Escolham uma funcionalidade simples (ex: login, cadastro, pesquisa)
2. Definam ID e título claros
3. Escrevam uma descrição completa
4. Adicionem pelo menos 3 critérios de aceitação
5. Utilizem a sintaxe Markdown correta

Tempo: 15 minutos

Referências

Sommerville, I. (2019). *Engenharia de Software.* 10ª ed. Pearson.

Pressman, R. S., & Maxim, B. R. (2021). *Engenharia de Software: Uma Abordagem Profissional.* 9ª ed. McGraw Hill.

Wiegers, K. & Beatty, J. (2013). *Software Requirements.* 3ª ed. Microsoft Press.