### Tema 1 – Apresentação

O que realmente o sistema deve fazer; quais são as entradas disponíveis e as saídas esperadas para o sistema, ou seja, que os objetivos estão alinhados com a necessidade?

### Tema 2 - Introdução a Requisitos de Sistemas

#### Módulo 1 – Introdução:

Imagine que uma padaria quer um programa que registre automaticamente a entrada e saída de produtos, como pão, leite e ovos. Assim, ao vender ou receber mercadorias, o sistema atualiza tudo sozinho.

#### Módulo 2 – Análise de Dados:

Antes de criar o programa, é importante entender as necessidades da padaria. Por exemplo, se não soubermos que o dono quer ser avisado quando o estoque de farinha estiver baixo, o sistema pode acabar sendo inútil.

#### Módulo 3 – Levantamento de Requisitos:

O programador conversa com o dono da padaria para entender o que ele precisa. Assim, descobre que o programa deve emitir alertas quando os produtos estiverem acabando e gerar relatórios de vendas diárias.

#### Módulo 4 – Qualidade do Software:

Durante a criação do programa, o desenvolvedor testa várias vezes para garantir que ele funciona bem. Por exemplo, verifica se o alerta de estoque baixo aparece na hora certa.

#### Módulo 5 – Controle e Garantia de Qualidade:

Antes de entregar o programa, o dono da padaria e o programador fazem testes juntos para confirmar que tudo funciona como esperado. Se algo estiver errado, dá tempo de corrigir.

#### Módulo 6 – Gerenciamento da Qualidade:

O programador cria um plano para garantir que o programa será feito de forma organizada, desde o início até a entrega, sempre checando se está dentro do combinado.

### Tema 3 – Níveis de Requisitos: Requisitos de Usuários e Requisitos de Sistemas

#### Módulo 1 – Introdução

Notadamente o sistema a ser desenvolvido substituirá ou aperfeiçoará algum outro existente ou automatizar um processo que atualmente não está sendo realizado por computador.

#### Módulo 3 – Requisitos no nível do usuário

*O que* o usuário quer? Ex.: visualizar saldo de conta.

#### Módulo 4 – Requisitos de sistema

*Como* o sistema vai realizar isso? Ex.: acessar a base de dados via HTTPS.

### Tema 4 – Tipos de Requisitos: Funcionais, Não Funcionais e Regras de Negócios

#### Módulo 1 – Requisitos Funcionais

são aqueles que estabelecem e descrevem as funcionalidades do sistema

Sistema deve possuir um módulo de saque que será disponibilizado em uma tela separada, essa será exibida conforme o interesse do cliente.

#### Módulo 2 – Requisitos não Funcionais

Um app de banco deve ter um requisito não funcional de confiabilidade e tempo, pois trabalha com dados sensíveis (saldo de conta) e deve exibir a tela principal em no máximo 3 segundos.

Requisitos de produto, requisitos organizacionais e requisitos externos etc.

MÉTRICAS PARA ESPECIFICAR REQUISITOS NÃO FUNCIONAIS

Velocidade, tamanho, facilidade de uso, confiabilidade, robustex, portabilidade (número de sistemas alvo).

### Tema 5 - Identificação de stakeholders

#### Módulo 1 – Stakeholders

Os stakeholders são todas as pessoas ou grupos que têm interesse ou influência no projeto ou serão impactados por ele. Eles desempenham um papel fundamental no levantamento de requisitos, pois ajudam a definir as necessidades e expectativas.

STAKEHOLDERS INTERNOS:

Equipe de projeto (desenvolvedores, analistas, designers).

Gestores e patrocinadores.

Colaboradores da empresa (ex.: suporte técnico, marketing).

STAKEHOLDERS EXTERNOS:

Usuários finais (clientes).

Fornecedores e parceiros de negócios.

Órgãos reguladores ou governos.

STAKEHOLDERS DIRETOS:

Envolvidos diretamente no uso ou desenvolvimento do produto.

STAKEHOLDERS INDIRETOS:

Impactados pelos resultados do projeto, mas sem envolvimento direto.

#### Módulo 2 – Identificando Stakeholders

MAPEAMENTO:

Liste todas as partes interessadas com base em seus interesses no projeto.

ANÁLISE DE PODER E INTERESSE:

Classifique stakeholders de acordo com sua influência e interesse no projeto:

ENTREVISTAS:

Converse com stakeholders identificados para validar seu papel e interesses.

#### Módulo 3 – Comunicação

STAKEHOLDERS SÃO A FONTE:

Eles desempenham o papel principal de fornecerem dados e informações sobre as necessidades e problemas que o projeto deve resolver.

GERENCIAMENTO DE CONFLITOS:

É comum que diferentes stakeholders tenham prioridades conflitantes. Cabe à equipe de projeto negociar e alinhar expectativas.

COMUNICAÇÃO CONTÍNUA:

Manter os stakeholders informados durante todo o ciclo de vida do projeto ajuda a evitar problemas futuros e promove alinhamento.

#### Módulo 4 – Levantamento de Requisitos

O levantamento de requisitos é o processo de identificar, coletar e documentar as necessidades e expectativas relacionadas ao projeto ou produto. Ele serve como base para garantir que o produto final atenda às expectativas dos usuários e das partes interessadas.

ENTREVISTAS:

Conversas com stakeholders para entender suas necessidades.

Exemplos: gestores, usuários finais, equipe técnica.

WORKSHOPS:

Sessões colaborativas com múltiplos stakeholders para debater e priorizar requisitos.

BRAINSTORMING:

Discussões abertas para coletar ideias e necessidades.

QUESTIONÁRIOS E PESQUISAS:

Coleta de informações de um público maior, como usuários potenciais.

OBSERVAÇÃO:

Analisar como os usuários realizam suas atividades para identificar problemas e necessidades.

ESTUDO DE DOCUMENTAÇÃO:

Análise de sistemas existentes, manuais e relatórios.

PROTOTIPAGEM:

Criação de modelos visuais (wireframes ou mockups) para validar requisitos.

#### Módulo 5 – O que é JAD?

O JAD (Desenvolvimento Conjunto de Aplicações) é um processo estruturado que reúne stakeholders, usuários finais, analistas e outros participantes-chave para realizar um trabalho colaborativo e intensivo, geralmente em workshops, para:

#### Módulo 6 – Ferramentas Úteis no JAD

Mapas mentais (para estruturar ideias).

Quadros brancos virtuais (ex.: Miro, MURAL, Jamboard).

Software de prototipagem (ex.: Figma, Balsamiq).

Documentos colaborativos (ex.: Google Docs, Confluence).

### Tema 6 – Documentação de Requisitos de Software

Documento de requisitos de software ou Especificação de Requisitos de Software (SRS – do inglês Software Requeriments Specification)

#### Módulo 1 – Essencialidade em Síntese

1. CENTRALIZA O CONHECIMENTO

**Cenário**: O gerente da clínica tem em mente que o sistema deve permitir a visualização da agenda de cada médico em tempo real.

**Problema** **Evitado**: Se essa informação não for documentada e o gerente sair de férias ou deixar o projeto, a equipe de desenvolvimento pode ignorar essa necessidade.

**Solução**: Documentar o requisito ("O sistema deve permitir que os médicos acessem sua agenda em tempo real.") garante que a equipe tenha acesso à informação.

2. PROMOVE A TRANSPARÊNCIA

**Cenário**: O médico quer ter controle sobre os horários disponíveis, enquanto a recepcionista prefere que os horários sejam abertos automaticamente.

**Problema** **Evitado**: Conflitos entre os stakeholders sobre como o sistema deve funcionar.

**Solução**: A documentação detalha como o sistema conciliará essas demandas, garantindo que todos saibam qual será o funcionamento final.

3. FACILITA A COMUNICAÇÃO

**Cenário**: Um desenvolvedor entende que o sistema deve apenas enviar e-mails de confirmação, enquanto o gestor esperava notificações via SMS também.

**Problema** **Evitado**: Mal-entendidos que poderiam gerar uma solução incompleta.

**Solução**: Documentar o requisito ("Enviar notificações por e-mail e SMS para cada agendamento.") deixa claro o escopo.

4. APOIA A RASTREABILIDADE

**Cenário**: Durante o desenvolvimento, percebe-se que a funcionalidade de "cancelamento de consulta pelo paciente" foi ignorada.

**Problema** **Evitado**: Dificuldade de identificar quem solicitou essa funcionalidade e quando ela foi acordada.

**Solução**: A documentação permite rastrear que o requisito foi definido na primeira reunião com os médicos.

5. EVITA RETRABALHO

**Cenário**: Após a entrega do sistema, a equipe descobre que a funcionalidade de restrição de horários (para pausas e intervalos médicos) não foi considerada.

**Problema** **Evitado**: Redesenho do sistema, o que geraria custos adicionais e atrasos.

**Solução**: Documentar o requisito desde o início garante que ele seja implementado na primeira versão.

#### Módulo 2 – Perfil de um Documento

Então agora, mais contextualizados sobre o perfil de um documento de requisitos, passaremos para uma nova etapa, na qual iremos aprender detalhes sobre a sua estrutura, conhecendo o que pode ser considerado para a sua criação.

Para tal finalidade, utilizaremos a composição definido por Sommerville (2009) – bibliografia básica de nossa disciplina, a qual está disposta abaixo:

**Prefácio** - Deve definir os possíveis leitores do documento e descrever seu histórico de versões, incluindo uma justificativa para a criação de uma nova versão e um resumo das mudanças feitas em cada versão.

**Introdução**- Deve descrever a necessidade para o sistema. Deve descrever brevemente as funções do sistema e explicar como ele vai funciona com outros sistemas. Também deve descrever como o sistema atende aos objetivos globais do negócio ou estratégicos da organização que encomendou o software.

**Glossário** - Deve definir os termos técnicos usados no documento. Não deve conter suposições sobre o conhecimento ou experiência do leitor.

**Definição de requisitos de usuário** - USUÁRIO: Deve descrever os serviços oferecidos ao usuário. Os requisitos não funcionais do sistema também devem ser descritos nessa seção. Essa descrição pode usar linguagem natural, diagramas ou outras notações compreensíveis para os clientes. Normas produtos que devem ser seguidos devem sem especificados.

**Modelos do sistema** - Pode incluir modelos gráficos do sistema que mostram relacionamentos entre os componentes do sistema, o sistema e seu ambiente.

**Evolução do sistema** - Deve descrever os pressupostos fundamentais em que o sistema se baseia, bem como quaisquer mudanças previstas, em decorrência da evolução do hardware, de mudanças nas necessidades do usuário, etc. Essa seção é útil para projetistas de sistemas, pois pode ajudá-los a evitar decisões capazes de restringir possíveis mudanças futuras no sistema.

**Apêndices** - Deve fornecer informações detalhadas e especificas em relação à aplicação em desenvolvimento, além das descrições de hardware e banco de dados, por exemplo. Os requisitos de hardware definem as configurações mínimas do sistema. Requisitos de banco de dados definem a organização lógica dos dados usados pelo sistema e os relacionamentos entre esses dados.

**Índice** - Vários índices podem ser incluídos no documento. Pode haver, além de um índice alfabético normal, um índice de diagramas, de funções, dentre outros que sejam pertinentes.

Atenção: Lembre-se que essa proposta não é única e definitiva, podendo sofrer adaptações, inclusões e exclusões, a depender da necessidade do cliente dos responsáveis pela sua criação.

### Tema 7 – Engenharia de Requisitos e Estudos de Viabilidade

#### Módulo 1 – Engenharia de Requisitos na Prática

SISTEMA DE PONTO ELETRÔNICO

**Levantamento de Requisitos**

→ Coleta das necessidades dos stakeholders.

* O RH quer um relatório diário de registros.
* Os funcionários querem acessar seus horários pelo celular.
* A equipe de TI sugere que a solução seja integrada ao sistema atual.

**Análise de Requisitos**

→ Refinamento e priorização dos requisitos.

* Após o levantamento, foram definidos os requisitos mais importantes:

✅ O sistema deve ter login por biometria.

✅ O sistema deve gerar relatórios diários para o RH.

⚠️ A funcionalidade de alertas por e-mail foi considerada opcional e será implementada depois.

**Especificação de Requisitos**

→ Documentação formal dos requisitos.

* REQ001: O sistema deve permitir login via biometria ou senha.
* REQ002: O sistema deve registrar horários de entrada e saída automaticamente.
* REQ003: O sistema deve gerar relatórios em PDF para o RH.

**Validação de Requisitos**

→ Confirmação de que os requisitos atendem às expectativas.

Antes de iniciar o desenvolvimento, um protótipo da tela de login é apresentado ao RH.

* O RH confirma que o modelo está adequado.
* Os funcionários pedem uma opção extra de esqueci minha senha.
* O requisito é ajustado antes do desenvolvimento.

**Gerenciamento de Requisitos**

→ Monitoramento e controle dos requisitos durante o projeto.

Durante os testes, descobre-se que algumas versões antigas de celulares não suportam biometria.

🔄 A equipe adiciona um novo requisito:

REQ004: Se o celular não suportar biometria, o sistema deve permitir login por código SMS.

#### Módulo 2 – Viabilidade

Um hospital deseja desenvolver um sistema de controle de acesso eletrônico para restringir o acesso a áreas específicas.

1. **Viabilidade Técnica**

O hospital já possui rede Wi-Fi suficiente para um sistema baseado em RFID?

✅ Sim

A equipe de TI pode dar suporte ao novo sistema?

✅ Sim

1. **Viabilidade Econômica**

O custo do projeto é de R$ 500.000,00 e o hospital tem orçamento para isso?

✅ Sim

1. **Viabilidade Operacional**

Os funcionários conseguem se adaptar rapidamente ao novo sistema?

⚠️ Pode ser necessário treinamento.

1. **Viabilidade Legal**

A coleta de dados dos funcionários e visitantes respeita as normas da LGPD?

✅ Sim, se os dados forem protegidos.

1. **Viabilidade de Cronograma**

O sistema precisa estar pronto em 3 meses. A empresa fornecedora consegue entregar no prazo? ⚠️ Pode ser necessário reduzir o escopo inicial.

**➡ Decisão Final:** O projeto é viável, mas precisa de ajustes no treinamento da equipe e no cronograma.

### Tema 8 - Elicitação De Requisitos

#### Módulo 1 – Definições

LEVANTAMENTO – o cliente fornece informações e faz solicitações de requisitos que ele considera relevante.

ELICITAÇÃO – a equipe de desenvolvimento incentiva e ajuda o cliente a refletir e investigar requisitos ainda não considerados.

#### Módulo 2 – EXEMPLO

A elicitação de requisitos realmente vai **além do óbvio** e busca descobrir oportunidades e necessidades que o cliente ou usuário final pode não ter identificado.

**Levantamento de Requisitos (Básico e Objetivo)**

Aqui, o analista apenas coleta informações diretas sobre o sistema:

✅ Controle de abastecimento por tipo de combustível.

✅ Aceitação de diferentes formas de pagamento (cartão, dinheiro, PIX).

✅ Emissão de nota fiscal e integração com sistemas fiscais.

✅ Controle de estoque de combustíveis e lubrificantes.

✅ Relatórios financeiros para gestão.

**Elicitação de Requisitos (Explorando Novas Necessidades e Inovações)**

🔹 SISTEMA AVANÇADO PARA UM POSTO DE COMBUSTÍVEL:

---

**Os alvos a serem alcançados**

📌 Objetivo: Melhorar a eficiência e fidelizar clientes.

📌 Exemplo: Implementar um programa de pontos onde clientes acumulam benefícios a cada abastecimento, incentivando a recorrência.

---

**Os pontos a serem acompanhados**

📌 Monitoramento: Indicadores de desempenho do sistema.

📌 Exemplo: O sistema deve gerar relatórios automáticos de consumo e estoque, alertando gestores sobre a necessidade de reabastecimento antes do combustível acabar.

---

**Como se encaixa no contexto das necessidades do negócio**

📌 Integração com estratégia empresarial.

📌 Exemplo: Criar um módulo de parcerias com frotas e aplicativos de transporte, permitindo abastecimento com cobrança direta e descontos exclusivos para motoristas parceiros.

---

**O comportamento ou operacionalização da solução na rotina da empresa**

📌 Uso prático e impacto operacional.

📌 Exemplo: Funcionários utilizam tablets para processar pagamentos e liberar bombas remotamente, reduzindo filas e otimizando o atendimento.

---

**💡 Resumo:** A elicitação de requisitos não se limita ao que o sistema deve fazer, mas como ele atinge objetivos estratégicos, monitora métricas-chave, se alinha ao negócio e opera no dia a dia da empresa.

#### Módulo 3 – Complexidade

**Mas então porque se apresenta como um processo extremamente complexo?**

**Problemas de escopo:** excesso ou falta de detalhamento. Os clientes/usuários desconhecem o que é importante (ou até mesmo quer ocultar), inibindo os limites do sistema, o que dificulta uma definição completa.

**Problemas de compreensão:**

omitem informações que julgam óbvias; clientes/usuários desconhecem ou estão em dúvidas sobre as necessidades e como seu papel é fundamental; é leigo ou limitado no conhecimento de seu ambiente computacional ou do domínio do seu negócio e etc.

**Problemas de volatilidade:** mudanças constantes nos requisitos.

#### Módulo 4 – Plano de Ação

A partir deste cenário, algumas **ações são sugeridas** para superar estes problemas, a fim de uma abordagem organizada para o processo da elicitação. São eles:

---

**Identificar todos os stakeholders e incluir seus preconceitos organizacionais**

📌 **Stakeholders:** Donos do posto, funcionários, fornecedores de combustível, clientes, empresas parceiras (frotas, aplicativos de transporte), órgãos reguladores.

📌 **Preconceitos organizacionais:**

- Donos podem resistir a mudanças tecnológicas por custo elevado.

- Funcionários podem temer que a automação reduza empregos.

- Clientes podem desconfiar da segurança do pagamento digital.

---

**Definir o ambiente técnico no qual o sistema será instalado**

📌 **Exemplo:** O sistema será instalado em servidores locais com backup na nuvem, terá integração com as bombas de combustível, terminais de pagamento e sistemas fiscais, e será acessado via tablets, computadores e dispositivos móveis.

---

**Ter domínio sobre o que é o sistema e o que ele realmente representa**

📌 **Exemplo:** O sistema não é apenas um software de controle de vendas, mas sim uma plataforma estratégica para otimizar a gestão, reduzir desperdícios, melhorar a experiência do cliente e aumentar a competitividade do posto.

**Incentivar a participação de todos os stakeholders**

📌 **Exemplo:** Realizar reuniões com funcionários e gestores, coletar sugestões de motoristas via pesquisas de satisfação e testar o sistema com empresas parceiras, garantindo que todas as partes interessadas sejam ouvidas antes da implementação final.

**💡 Conclusão:** A elicitação permite criar um sistema diferenciado, que não apenas atende às necessidades atuais do posto, mas gera **valor e vantagens competitivas no mercado.**

### Tema 9 – Validação De Requisitos

Nessa **etapa**, os três elementos principais – **documento de requisitos, protótipo (rascunho) e testes** – são analisados para garantir que o sistema atenda às necessidades do cliente e dos stakeholders antes da **implementação final**.

Dessa forma, detectar um erro em **fases finalistas** de um projeto, pode chegar a ser danoso a ponto de incompatibilizar a continuidade, visto que pode ser tão custosa que não existe recursos para comportá-la. Por isso, a atenção deve ser constante e minuciosa.

PROTOTIPAGEM

Um protótipo de um sistema é uma representação rudimentar ou detalhada, visual ou funcional inicial do software, criada para testar conceitos, validar requisitos e coletar feedback antes do desenvolvimento completo. Ele pode variar de simples wireframes até protótipos interativos e navegáveis.

#### Módulo 1 – Cenários Problemáticos:

**🔹 Durante a Elaboração do Documento de Requisitos**

**📌 Problemas identificados:**

- Descrição pouco clara dos requisitos

- Ambiguidade entre requisitos

- Conflitos entre requisitos

- Ausência de informação

**📌 Exemplo:** Durante uma revisão do documento, percebe-se que não há informações sobre a política de garantias da assistência técnica. Esse erro pode ser corrigido antes de iniciar o desenvolvimento.

---

**🔹 Durante a Prototipagem**

**📌 Problemas identificados:**

- Não conformidade com normas

- Falhas na modelagem dos requisitos

- Ambiguidade entre requisitos

**📌 Exemplo:** No protótipo do sistema, percebe-se que não há um campo para registrar o status do conserto, o que pode comprometer o acompanhamento pelos clientes. Esse erro é ajustado antes de passar para o desenvolvimento.

---

**🔹 Durante os Testes**

**📌 Problemas identificados:**

- Falhas na modelagem dos requisitos

- Requisitos não realistas

**📌 Exemplo:** Durante testes com usuários, percebe-se que a função de diagnóstico automático do smartphone é inviável, pois depende de hardware específico. O requisito precisa ser reformulado.

---

**🔹 Após a Implementação (Pior Cenário)**

**📌 Problemas identificados:**

- Conflitos entre requisitos

- Não conformidade com normas

- Ausência de informação

**📌 Exemplo:** O sistema é lançado, mas os clientes reclamam que não conseguem visualizar o status do conserto online, pois a funcionalidade não foi corretamente implementada. Agora, uma nova versão precisa ser desenvolvida, aumentando o custo do projeto.

---

**💡 Conclusão**

Quanto mais cedo os erros forem detectados, **menor será o impacto no projeto**. A **validação contínua** dos requisitos durante todas as fases (documentação, prototipagem, testes) reduz falhas e evita problemas graves após a implementação.

#### Módulo 2 – Exemplo de Validação Contínua

A validação contínua ocorre ao longo de todo o ciclo de desenvolvimento, garantindo que cada etapa do projeto esteja alinhada com os requisitos e expectativas dos stakeholders.

**📌 Exemplo:** No desenvolvimento do sistema para assistência técnica de smartphones, a equipe **realiza revisões semanais** com os técnicos e gerentes da loja para validar se:

* O sistema está registrando corretamente os consertos.
* As notificações aos clientes funcionam conforme esperado.
* A interface é intuitiva para os funcionários.

A cada nova funcionalidade implementada, feedback é coletado e ajustes são feitos antes de seguir para a próxima fase, evitando retrabalho e garantindo aderência às necessidades do usuário.

#### Módulo 3 – Exemplo de Validação Simultânea

A validação simultânea ocorre quando diferentes partes do sistema são testadas e validadas ao mesmo tempo, geralmente por diferentes grupos de stakeholders.

**📌 Exemplo:** Durante a fase de testes do sistema para assistência técnica:

Os técnicos testam o fluxo de registro e finalização de consertos.

Os clientes acessam a versão beta do portal para verificar se conseguem acompanhar o status do reparo.

Os gerentes avaliam os relatórios gerados pelo sistema para garantir que atendem às necessidades do negócio.

Isso permite que feedbacks de diferentes áreas sejam coletados ao mesmo tempo, acelerando a validação e garantindo que o sistema funcione corretamente para todos os usuários antes da implementação final.

### Tema 10 – Gerenciamento de requisitos

A **evolução dos requisitos de sistema** ocorre de forma constante, sofrendo **mutações** conforme novas percepções surgem e necessitando **gerenciamento**.

Nesse aspecto, cada requisito deve possuir um **identificador**, ou melhor, pode ser definida uma **política para compor cada identificação**. Ela precisa ser única e mesmo que o requisito deixe de ser utilizado, deve mantê-la para fins de **histórico**.

Portanto, o gerenciamento de requisitos, de certa forma, pode ser visto como **capturar uma "foto"** do que um sistema ou produto deve ser, antes de começar o processo de desenvolvimento ou mudanças. Assim como tirar uma foto de um dispositivo antes de desmontá-lo, você está documentando todos os componentes essenciais e **detalhes que precisam ser preservados** ou levados em conta ao longo do processo.

#### Módulo 1 – Evolução

🎮 SISTEMA DE UM JOGO DE AVENTURA

**🔹 Compreensão Inicial do Problema:**

Criar um jogo de aventura com mundo aberto, onde o jogador explora cenários e combate inimigos.

**🔹 Requisitos Iniciais:**

✅ Mundo aberto com mapa extenso.

✅ Sistema de combate com armas e magias.

✅ Missões principais e secundárias.

✅ NPCs interativos para dar informações e quests.

**🔹 Compreensão Alterada do Problema:**

Os testes iniciais mostraram que o mundo aberto estava vazio e cansativo, reduzindo a imersão. Os jogadores também queriam mais impacto nas escolhas da história.

**🔹 Requisitos Alterados:**

✅ Redução do tamanho do mapa, focando em áreas densas e interativas.

✅ Introdução de eventos dinâmicos e NPCs com inteligência adaptável.

✅ Sistema de escolhas que alteram a narrativa, impactando o mundo do jogo.

**💡 Conclusão:** A evolução dos requisitos ajudou a transformar o jogo em uma experiência mais envolvente e imersiva.

#### Módulo 2 – Mudanças

**Estrutura de Requisitos com Adaptabilidade a Mudanças**

Para permitir mudanças nos requisitos sem comprometer o projeto, pode-se utilizar:

**✅ Requisitos modulares:** cada funcionalidade é independente e pode ser ajustada sem afetar todo o sistema.

**✅ Versões de requisitos**: cada atualização é documentada para manter o histórico das mudanças.

**✅ Backlog priorizado:** funcionalidades são organizadas por importância e impacto, facilitando ajustes.

**🔹 Exemplo prático:**

* Inicialmente, o sistema de IA dos NPCs previa apenas falas predefinidas.
* Após testes, foi identificado que os NPCs precisavam de respostas dinâmicas baseadas nas escolhas do jogador.
* O requisito foi atualizado sem impactar o restante do jogo, pois o sistema foi projetado modularmente.

**Vínculos de Rastreabilidade para Representar Dependências entre Requisitos**

Os requisitos de um jogo podem ter dependências entre si. Para rastrear essas ligações, pode-se usar:

**✅ Matriz de rastreabilidade** (requisitos → funcionalidades → código → testes).

✅ **Ferramentas de ALM (Application Lifecycle Management)** como Jira, Trello ou Azure DevOps.

✅ **Hierarquia de requisitos:** principais (macro) e derivados (dependentes).

**🔹 Exemplo prático:**

* **Requisito 1:** O jogador pode escolher entre diferentes facções.
* **Requisito 2 (dependente):** As missões e aliados do jogador mudam conforme a facção escolhida.
* **Requisito 3 (dependente):** O final do jogo é alterado com base nas escolhas de facção.

Se Requisito 1 mudar (ex.: facções removidas ou reformuladas), Requisitos 2 e 3 precisam ser ajustados.

**Estabelecer Rastreabilidade entre Itens Correlatos**

Criar links entre requisitos permite entender o impacto das mudanças. Pode ser feito com:

✅ **Rastreabilidade para requisitos funcionais e não funcionais** (ex.: desempenho, compatibilidade).

**✅ Relacionamento** entre regras de jogo e requisitos técnicos.

✅ **Rastreamento de impacto** antes de modificar um requisito.

**🔹 Exemplo prático:**

* **Requisito Funcional:** Implementar sistema de combate com magias.
* **Requisito Técnico Correlato:** Criar shaders e partículas para efeitos mágicos.
* **Requisito de Desempenho:** O sistema de magias deve manter FPS acima de 60.

Se houver mudanças no requisito de magias, os efeitos gráficos e a otimização de desempenho também precisarão ser ajustados.

**💡 Conclusão**

A **rastreabilidade** e a **adaptabilidade** são essenciais para **gerenciar mudanças** nos requisitos sem comprometer a qualidade do jogo. Isso garante que cada atualização seja controlada e suas dependências sejam ajustadas corretamente. 🚀

### Tema 11 - Casos de uso

Uma modelagem de um determinado sistema é um processo que consiste na representação de uma **visão** (ou perspectiva) do que se espera do sistema, no tocante ao seu **funcionamento** e resultado(s). É um consenso que ter uma representação visual de seu sistema **antes que ele entre na etapa de implementação** é de fundamental importância.

Desde meados da década de 80, o CASO DE USO estabelece uma metodologia que institui regras para a modelagem de sistemas. É como se fosse a **planta baixa de uma residência.** Antes que venhamos a construí-la, precisamos observar o desenho e demais informações, para evitarmos desagradáveis surpresas no o produto depois de terminado, ou seja, de nossa casa, por exemplo.

Representados por **diagramas\***, os Casos de Uso têm o objetivo de auxiliar a comunicação entre os analistas e o cliente. Ele então expõe uma espécie de **cenário** que mostra as funcionalidades do sistema do ponto de vista do usuário. Enfim, o cliente deve ter acesso através do diagrama de Casos de Uso a identificação das principais funcionalidades de seu sistema.

\* diagrama é uma representação gráfica, ou seja, através de desenhos.