Aula 2: Definição de Escopo e Levantamento de Requisitos

---

1. Introdução ao Processo de Levantamento de Requisitos

Desenvolver um sistema corporativo de qualidade vai muito além de programar: é preciso entender, detalhar e organizar *o que* deve ser construído, *por que* e *para quem*. É nesta etapa que se define **o escopo do sistema**, ou seja, tudo aquilo que ele deverá (ou não) fazer. A clareza nesse momento é fundamental para evitar retrabalho, conflitos de expectativa e desperdício de tempo.

Um projeto bem-sucedido nasce de uma boa definição de requisitos. É preciso conversar com futuros usuários, entender processos reais, analisar soluções existentes e transformar tudo isso em funcionalidades e regras bem descritas. O resultado será um guia seguro para todas as próximas etapas do projeto.

---

2. Técnicas para Levantar e Especificar Requisitos

2.1. Brainstorming

O brainstorming é uma dinâmica essencial para o início de qualquer projeto. Reúna a equipe e incentive todos a sugerirem, livremente, funcionalidades, ideias de telas, regras e até mesmo problemas comuns enfrentados em sistemas de eventos. Não existe certo ou errado nesta etapa: o objetivo é gerar o maior número possível de ideias, sem julgamentos.

2.2. Entrevistas Simuladas

Simule entrevistas em sala: um aluno (ou o professor) assume o papel de *cliente/usuário* do sistema, enquanto os demais fazem perguntas para descobrir necessidades e desejos. Perguntas como:

- “Quais tarefas você precisa automatizar?”
- “Quais são os maiores problemas dos eventos hoje?”
- “Que informações são essenciais em um cadastro de evento?”
- “Como você faz o controle de inscrições?”

2.3. Análise de Sistemas Similares

Busque exemplos de sistemas de gestão de eventos disponíveis no mercado ou open-source. Analise suas principais funcionalidades, telas, fluxos e tente extrair ideias para o seu projeto. Use isso como referência para não reinventar a roda e para identificar funcionalidades consideradas “básicas” pelo mercado.

2.4. Técnicas Complementares

- **Workshops com usuários fictícios**
- **Mapeamento de jornadas** (roteiro: desde o cadastro de evento até o feedback do participante)

- **Lista de dores e desejos** (o que incomoda e o que seria “sonho” em uma plataforma de eventos)

---

3. Requisitos Funcionais e Não-Funcionais

Após levantar ideias, é hora de organizar tudo em **requisitos**. Eles serão divididos em dois grandes grupos:

3.1. Requisitos Funcionais

São as funcionalidades que o sistema *deve* oferecer. Devem ser claros, objetivos e sem ambiguidade. Exemplos para o sistema de eventos:

- Permitir que organizadores cadastrem, editem e excluam eventos.
- Permitir que participantes busquem eventos e visualizem detalhes.
- Gerenciar inscrições de participantes nos eventos.
- Processar pagamentos de inscrições, quando aplicável.
- Enviar confirmações e recibos por e-mail.
- Coletar e armazenar feedback dos participantes.
- Gerar relatórios simples de eventos e inscrições.

3.2. Requisitos Não-Funcionais

Tratam das *qualidades* do sistema, e não de suas funções. Envolvem desempenho, segurança, disponibilidade, usabilidade, entre outros. Exemplos:

- O sistema deve suportar, ao menos, 1.000 usuários simultâneos.
- Todas as operações críticas devem ser protegidas por autenticação.
- A plataforma deve ser responsiva (funcionar bem em desktop e dispositivos móveis).
- As informações de pagamento devem ser transmitidas de forma segura.
- A recuperação de falhas deve ser rápida e o sistema deve registrar erros para posterior análise.

---

4. User Stories: Dando Vida aos Requisitos

Uma maneira didática e prática de descrever funcionalidades é através das **user stories**. Elas apresentam os requisitos do ponto de vista de quem irá utilizar o sistema. A estrutura clássica é:

> “Como [ator/usuário], quero [realizar ação] para [obter benefício].”

Exemplos práticos para nosso projeto:

- Como organizador, quero criar um novo evento informando nome, data, local e limite de vagas para começar a divulgar.
- Como participante, quero me inscrever rapidamente em um evento pelo celular para garantir minha vaga.
- Como organizador, quero visualizar uma lista de participantes para gerenciar o acesso no dia do evento.
- Como participante, quero receber um recibo por e-mail após o pagamento da inscrição para comprovar minha participação.

- Como organizador, quero acessar feedbacks dos participantes para melhorar futuros eventos.

---

5. Identificação de Atores e Papéis

Para estruturar o sistema, precisamos identificar *quem* irá interagir com ele. Exemplos típicos:

- **Organizador/Admin:** Responsável por cadastrar e gerenciar eventos, acessar relatórios, visualizar feedbacks.
- **Participante:** Realiza inscrições, visualiza eventos, faz pagamentos, envia feedbacks.
- **Sistema/Serviços externos:** Integração para envio de e-mails, processadores de pagamento, entre outros.

---

6. Documentação do Escopo e Requisitos

O próximo passo é consolidar tudo em um **Documento de Visão e Escopo**. Essa documentação deve conter:

- **Título do Projeto**
- **Descrição do Problema:** O que o sistema pretende resolver? (ex.: dificuldades em gerenciar eventos corporativos, inscrições manuais, pagamentos confusos, etc.)
- **Objetivo do Sistema:** Ajudar organizadores a criar e gerenciar eventos e participantes a se inscreverem com facilidade e segurança.
- **Atores Identificados:** Lista dos perfis envolvidos e seus principais objetivos.
- **Requisitos Funcionais:** Listagem numerada e detalhada das principais funcionalidades.
- **Requisitos Não-Funcionais:** Lista das qualidades esperadas do sistema.
- **User Stories de Exemplo:** Pelo menos três user stories para ilustrar o funcionamento.
- **Divisão inicial em módulos/microserviços:** Uma proposta simples de como dividir o sistema, por exemplo:
- Serviço de Usuários/Auth
- Serviço de Eventos
- Serviço de Inscrições
- Serviço de Pagamentos
- Serviço de Notificações
- Serviço de Feedback

---

7. Considerações Finais

Definir o escopo e os requisitos é um passo estratégico. Um documento claro evita muitos problemas futuros, economiza tempo, aumenta a produtividade da equipe e potencializa o sucesso do projeto. Lembre-se: sistemas bem planejados nascem de requisitos bem definidos!

Próximos passos: com o documento de escopo validado, na próxima aula a equipe irá avançar para a **modelagem do domínio** e a identificação dos microserviços necessários — etapa essencial para o design da arquitetura do sistema.