Aula 16 – Definição do Problema: Job Stories e User Stories

Além da Funcionalidade: Descobrindo o "Porquê" por Trás das Ações do Usuário

Imagine passar meses, talvez anos, dedicando seu talento e energia para construir um produto digital incrível. O código é limpo, o design é elegante, mas, após o lançamento, o silêncio. Poucas pessoas usam, e as que usam, logo abandonam. Esse cenário, infelizmente comum, raramente acontece por falha na execução, mas sim por um erro fundamental no início do processo: a equipe construiu uma solução brilhante para o problema errado. Esta aula é sobre a arte do trabalho de detetive no design, sobre aprender a escutar o que os usuários não dizem e a entender suas verdadeiras motivações antes mesmo de desenhar a primeira tela.

Ao final desta aula, você não verá mais um pedido de "precisamos de um botão de login" da mesma forma. Você será capaz de ir além da solicitação superficial e articular a necessidade humana real por trás dela. Vamos equipá-lo com ferramentas para redigir declarações de problemas que funcionam como uma estrela-guia para toda a equipe, garantindo que o navio do projeto nunca perca o rumo. Exploraremos dois frameworks poderosos: o **Jobs to Be Done**, para focar na motivação, e as **User Stories**, para traduzir essa motivação em ações concretas.

Esta habilidade é um divisor de águas na sua carreira, seja você um futuro designer, desenvolvedor ou gerente de produto. É o que diferencia os construtores de funcionalidades dos criadores de soluções que as pessoas realmente "contratam" para melhorar suas vidas. Nesta jornada, vamos primeiro solidificar a importância de um bom diagnóstico — a declaração do problema. Em seguida, mudaremos nossa perspectiva com o "Jobs to Be Done". Por fim, aprenderemos a arte de escrever e organizar User Stories eficazes, preparando o terreno para a próxima fase do nosso trabalho: a priorização.

A Fundação: Por que uma Declaração de Problema Clara é Tudo

Você já tentou montar um móvel complexo sem o manual de instruções? Você tem todas as peças, todas as ferramentas, mas nenhuma direção clara. O resultado provável é uma estante bamba, tempo perdido e muita frustração. No desenvolvimento de produtos, iniciar um projeto sem uma declaração de problema clara e concisa é exatamente a mesma coisa. A equipe pode ser brilhante, as ferramentas podem ser de última geração, mas sem um entendimento compartilhado do "o quê" e do "porquê", o projeto está fadado a vagar sem rumo.

Uma **declaração de problema** (ou *Problem Statement*) não é uma lista de funcionalidades desejadas nem uma descrição da solução. É uma articulação focada e empática da barreira que impede um usuário de progredir. É o diagnóstico preciso que deve preceder qualquer prescrição. Muitas equipes, na ânsia de construir, pulam essa etapa crucial. Elas se apaixonam por uma ideia de solução — "vamos criar um aplicativo de agendamento!" — sem antes se apaixonarem pelo problema: a ansiedade de um paciente tentando marcar uma consulta, o tempo perdido em chamadas telefônicas, a incerteza sobre a disponibilidade do médico.

> 💡 **Exemplo de Declaração de Problema**
>
> "Profissionais ocupados precisam de uma maneira assíncrona para marcar compromissos, pois o modelo atual baseado em telefonemas consome tempo valioso e interrompe seu fluxo de trabalho"

Pense na declaração de problema como a fundação de um arranha-céu. Ninguém vê a fundação quando o prédio está pronto, mas é ela que garante que toda a estrutura permaneça de pé, estável e segura. Uma declaração de problema bem elaborada, como "Profissionais ocupados precisam de uma maneira assíncrona para marcar compromissos, pois o modelo atual baseado em telefonemas consome tempo valioso e interrompe seu fluxo de trabalho", serve como essa base sólida. Ela alinha a equipe, foca os esforços e se torna o critério pelo qual todas as decisões de design e desenvolvimento serão julgadas.

Mudando a Perspectiva com o "Jobs to Be Done" (JTBD)

Estamos acostumados a descrever os produtos pelo que eles fazem. Um aplicativo de streaming *toca música*. Uma furadeira *faz furos*. Essa visão, embora correta, é superficial e limitada. Ela nos prende ao produto e às suas funcionalidades, ofuscando a razão mais profunda pela qual alguém decide usar aquilo. E se, em vez de perguntar "o que seu produto faz?", nós perguntássemos "que 'trabalho' as pessoas estão 'contratando' seu produto para fazer?". Essa simples mudança de pergunta é a essência de um dos frameworks mais transformadores do design: o **Jobs to Be Done (JTBD)**.

Visão Tradicional

Foco no produto e suas funcionalidades

- O que o produto faz
- Características técnicas
- Recursos disponíveis

Visão JTBD

Foco no trabalho que precisa ser feito

- Por que as pessoas "contratam" o produto
- Contexto de uso
- Resultado desejado

A teoria, popularizada por Clayton Christensen, da Harvard Business School, propõe que os consumidores não compram produtos; eles os "contratam" para realizar um trabalho específico em suas vidas. Ninguém acorda pensando "eu quero comprar uma furadeira de 6mm". A pessoa pensa "eu preciso pendurar este quadro para que a sala fique mais aconchegante". O furo de 6mm é apenas um passo intermediário; o "trabalho" real é a transformação do ambiente. A furadeira é a solução contratada para esse trabalho.

> "A analogia clássica é a do milkshake." Uma pesquisa de uma rede de fast-food para aumentar as vendas de milkshake falhou porque perguntava aos clientes como poderiam "melhorar" o produto (mais doce? mais sabores?). Apenas quando um pesquisador observou o contexto é que a verdade apareceu: muitos milkshakes eram "contratados" por motoristas solitários no trajeto matinal para o trabalho. O "job" era ter algo interessante e que preenchesse o estômago durante um trajeto monótono.

Entender essa distinção é libertador. Os concorrentes do milkshake não eram outros milkshakes, mas bananas, donuts e outros itens fáceis de consumir no carro. O JTBD nos força a olhar para o contexto do usuário e para a concorrência real, que muitas vezes está fora da nossa categoria de produto.

A Anatomia de uma História de Trabalho (Job Story)

Compreender a filosofia do "Jobs to Be Done" é o primeiro passo. O segundo é traduzir essa filosofia em uma ferramenta prática que possamos usar no dia a dia do projeto. Se o framework é a lente através da qual vemos o mundo, a **Job Story** é a fotografia nítida que tiramos com essa lente. Ela captura não apenas a ação, mas a causa e o efeito, a motivação que impulsiona o usuário.

Quando

Situação/contexto que desencadeia a necessidade

Eu quero

Motivação/objetivo do usuário

Para que

Resultado esperado/benefício desejado

Diferente de outras formas de descrever requisitos, a Job Story coloca o foco principal no gatilho, na situação que desencadeia a necessidade. Sua estrutura é elegante e poderosa: "**Quando** [situação/contexto], **eu quero** [motivação/objetivo], **para que** [resultado esperado]". Cada parte desse quebra-cabeça é crucial. O "Quando" nos ancora no mundo real do usuário, o "Eu quero" revela a força motriz e o "Para que" nos mostra a visão de um futuro melhor que o usuário almeja.

> 📝 **Exemplo Prático**
>
> **Quando** eu recebo um lembrete para a minha consulta anual e estou no meio de um dia de reuniões, **eu quero** encontrar e agendar um horário compatível em menos de dois minutos, **para que** eu possa cuidar da minha saúde sem gerar mais estresse na minha agenda já lotada.

Vamos resgatar nosso exemplo do aplicativo de agendamento. Uma descrição focada em funcionalidade diria: "O usuário precisa de um calendário". Uma Job Story, no entanto, revela a história completa: "Quando eu recebo um lembrete para a minha consulta anual e estou no meio de um dia de reuniões, eu quero encontrar e agendar um horário compatível em menos de dois minutos, para que eu possa cuidar da minha saúde sem gerar mais estresse na minha agenda já lotada". A riqueza de detalhes nesse formato é imensa. Entendemos que a solução precisa ser rápida, eficiente e não intrusiva. A Job Story é a matéria-prima da empatia, fornecendo o "porquê" que inspira soluções verdadeiramente centradas no usuário.

Da Motivação à Ação: Construindo User Stories Eficazes

As Job Stories são a nossa ferramenta estratégica. Elas garantem que estamos apontando nosso navio para o destino certo, focados no problema real do usuário. No entanto, quando a equipe de desenvolvimento precisa começar a construir, ela necessita de instruções mais táticas, mais próximas da interação com o produto em si. É aqui que entra a parceira perfeita da Job Story: a **User Story**. Elas não são inimigas; são duas lentes diferentes para visualizar o mesmo desafio.

Job Story

Habita o "espaço do problema"

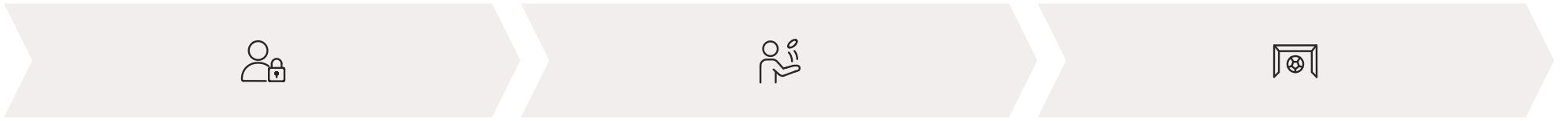
- Foca na motivação
- Independente da solução
- Estratégica
- Contexto e gatilho

User Story

Vive no "espaço da solução"

- Foca na funcionalidade
- Específica do produto
- Tática
- Ação e interação

Se a Job Story habita o "espaço do problema", a User Story vive no "espaço da solução". Ela traduz a motivação do usuário em uma funcionalidade específica dentro do produto que estamos criando. Ela serve como a unidade de trabalho fundamental em metodologias ágeis como o Scrum, representando uma pequena fatia de valor que pode ser entregue ao usuário. Ela cria um entendimento compartilhado sobre o que precisa ser construído, para quem e por quê.



Como um

Tipo de usuário/persona

Eu quero

Realizar uma ação

Para que

Alcançar um objetivo/benefício

A estrutura da User Story é provavelmente familiar para você, e sua simplicidade é sua força: "**Como um** [tipo de usuário/persona], **eu quero** [realizar uma ação], **para que** [eu possa alcançar um objetivo/benefício]". Essa fórmula garante que nunca percamos de vista três elementos críticos: o **público** (quem se beneficia), a **ação** (o que a interface deve permitir) e o **valor** (por que isso importa para o usuário). Se a Job Story é o destino final em nosso GPS ("chegar à praia para relaxar"), as User Stories são as instruções curva a curva ("vire à esquerda na próxima rua", "pegue a saída 25B"). Ambas são indispensáveis para uma viagem bem-sucedida.

O Padrão Ouro: Critérios INVEST para Boas User Stories

Escrever algo no formato "Como um..., eu quero..., para que..." é o primeiro passo, mas não garante que a história seja eficaz. Uma User Story mal formulada pode gerar mais dúvidas do que clareza, levando a retrabalho e frustração. Para evitar essa armadilha, a comunidade ágil desenvolveu um conjunto de critérios poderoso e fácil de lembrar, o acrônimo **INVEST**. Ele funciona como uma checklist de qualidade para garantir que nossas histórias sejam robustas e prontas para o desenvolvimento.

Independent

Cada história deve ser autocontida, na medida do possível, para que possa ser desenvolvida e entregue sem depender de outra.

Negotiable

A história não é um contrato fixo, mas um convite à colaboração entre o time de produto e os desenvolvedores para definir a melhor implementação.

Valuable

Deve entregar valor claro para o usuário final ou para o negócio. Se uma história não tem valor, por que a estamos construindo?

Estimable

A equipe de desenvolvimento precisa ter informações suficientes para fazer uma estimativa do esforço necessário para implementá-la.

Small

A história deve ser pequena o suficiente para ser concluída dentro de um único ciclo de desenvolvimento (sprint).

Testable

Devemos ser capazes de definir critérios claros para verificar se a história foi implementada corretamente.

> ❌ Exemplo Ruim: "Como usuário, eu quero gerenciar meu perfil"
>
> ✅ Exemplo Bom: "Como usuário registrado, eu quero alterar meu endereço de e-mail, para que eu possa receber notificações na minha conta atual"

Imagine uma história como: "Como usuário, eu quero gerenciar meu perfil". Ela falha em quase todos os critérios do INVEST: é grande, difícil de estimar e vaga demais para testar. A abordagem correta seria quebrá-la em histórias menores e mais focadas, como: "Como usuário registrado, eu quero alterar meu endereço de e-mail, para que eu possa receber notificações na minha conta atual". Esta sim, atende ao padrão ouro do INVEST.

Job Stories vs. User Stories: Uma Orquestra de Duas Peças

Até agora, pode parecer que estamos apresentando duas ferramentas que fazem a mesma coisa. É uma dúvida comum, mas a beleza desses frameworks está justamente em suas diferenças e na forma como se complementam. Tentar escolher entre Job Stories e User Stories é como pedir a um carpinteiro que escolha entre um martelo e uma serra. Ambas são essenciais, mas para momentos e propósitos distintos do trabalho. A questão não é "qual é a melhor?", mas "quando usar cada uma?".

A diferença fundamental reside no foco. A **Job Story** é obcecada pelo **problema**. Ela nos força a entender o contexto e a motivação do usuário, completamente agnóstica a qualquer solução ou produto. Ela é uma ferramenta de estratégia, usada nas fases iniciais de descoberta para garantir que estamos resolvendo uma necessidade real. Por outro lado, a **User Story** é focada na **solução**. Ela descreve uma interação específica que um usuário terá com o produto que estamos construindo. É uma ferramenta tática, usada no planejamento e na execução do desenvolvimento.

Pense na construção de uma ponte. A Job Story seria: "**Quando** as comunidades de duas cidades vizinhas precisam colaborar e trocar bens, **eu quero** uma forma de atravessar o rio de forma rápida e segura, **para que** possamos fortalecer nossa economia e laços sociais". Esta declaração não diz nada sobre uma ponte; a solução poderia ser uma balsa ou um túnel. Uma vez que decidimos pela ponte, surgem as User Stories: "**Como um** motorista de caminhão, **eu quero** faixas largas o suficiente, **para que** eu possa transportar minha carga sem risco de acidentes". A Job Story define a missão; as User Stories definem os detalhes da execução.

Critério	Job Story	User Story
Foco	O "porquê" (motivação e contexto)	O "o quê" (funcionalidade e interação)
Estrutura	Quando..., eu quero..., para que...	Como um..., eu quero..., para que...
Perspectiva	Independente da solução (espaço do problema)	Focada na interação com o produto (espaço da solução)
Uso Ideal	Fase de estratégia e descoberta	Fase de planejamento e desenvolvimento

Os Heróis Não Cantados: Critérios de Aceitação (ACs)

Uma User Story bem escrita, seguindo o padrão INVEST, nos dá uma excelente direção. Ela nos diz quem é o usuário, o que ele quer fazer e por quê. No entanto, ainda pode haver uma zona cinzenta de interpretação. O que exatamente significa "fazer login com sucesso"? E o que deve acontecer se a senha estiver errada? Para eliminar essa ambiguidade e garantir que todos na equipe — do designer ao testador — tenham uma definição compartilhada de "pronto", precisamos de um último elemento: os **Critérios de Aceitação**.

O que são ACs?

Os Critérios de Aceitação (ou *Acceptance Criteria* - ACs) são um conjunto de condições específicas e testáveis que a implementação de uma User Story deve satisfazer para ser considerada completa.

Formato Padrão

Geralmente são escritos no formato "Dado-Quando-Então" (*Given-When-Then*), que ajuda a descrever o cenário, a ação e o resultado esperado.

Função Principal

Eles são o "contrato" que detalha o comportamento esperado do sistema, transformando uma história potencialmente vaga em um requisito claro e verificável.

Vamos pegar a User Story: "Como um cliente recorrente, eu quero fazer login com meu e-mail e senha, para que eu possa acessar minha área de pedidos". Os ACs dariam vida a essa história:

📋 Exemplos de Critérios de Aceitação

Cenário 1 (Sucesso): Dado que estou na página de login, **quando** eu insiro meu e-mail e senha corretos e clico em "Entrar", **então** sou redirecionado para o meu painel de pedidos.

Cenário 2 (Falha): Dado que estou na página de login, **quando** eu insiro uma senha incorreta, **então** uma mensagem de erro "E-mail ou senha inválidos" é exibida abaixo do formulário.

Os Critérios de Aceitação são como as regras específicas de um jogo. A User Story nos diz o objetivo do jogo (marcar um gol), mas os ACs nos dizem as regras (a bola tem que cruzar a linha completamente, o jogador não pode estar em impedimento). Sem essas regras, é impossível saber se o objetivo foi realmente alcançado.

Juntando as Peças: O Product Backlog

Criamos Job Stories para entender a motivação, as quebramos em User Stories com o padrão INVEST e detalhamos cada uma com Critérios de Aceitação claros. Agora, temos um conjunto de requisitos bem definidos, mas onde eles devem morar? Deixá-los em documentos espalhados ou notas adesivas na parede é uma receita para o caos. Precisamos de um repositório central, uma fonte única da verdade para todo o trabalho a ser feito. Esse repositório é o **Product Backlog**.

Características do Backlog

- **Artefato vivo:** Constantemente atualizado
- **Dinâmico:** Itens podem ser adicionados, removidos ou modificados
- **Priorizado:** Ordenado por importância e urgência
- **Fonte única da verdade:** Repositório central de todo o trabalho

🍽️ Analogia do Restaurante

Pense no Product Backlog como o sistema de comandas de um restaurante movimentado. O chef (Product Owner) organiza as comandas por prioridade, e os cozinheiros sempre pegam a próxima do topo.

O Product Backlog é muito mais do que uma simples lista de tarefas. É um artefato vivo, dinâmico e priorizado de tudo o que é necessário para melhorar o produto. Ele contém todas as User Stories, mas também pode incluir tarefas técnicas, correção de bugs e trabalho de pesquisa. A característica mais importante do backlog é que ele é **ordenado**: os itens mais importantes e urgentes estão no topo, e os menos importantes estão no fundo. Essa ordem dita em que a equipe de desenvolvimento trabalhará a seguir.

Pense no Product Backlog como o sistema de comandas de um restaurante movimentado. Novos pedidos (User Stories) chegam constantemente da cozinha (time de produto). O chef (o Product Owner ou Gerente de Produto) organiza essas comandas na sua frente, colocando os pratos mais urgentes ou importantes primeiro. Ele pode reorganizar as comandas a qualquer momento se um cliente VIP chegar ou se um ingrediente estiver acabando. Nenhum cozinheiro pega uma comanda do meio da pilha; eles sempre pegam a próxima do topo. Essa organização garante que a cozinha esteja sempre trabalhando no que gera mais valor naquele exato momento.

Escrevendo Histórias para o Mundo de 2025: IA, Voz e Acessibilidade

Os princípios para escrever boas histórias são duradouros, mas o conteúdo dessas histórias precisa refletir o mundo em que vivemos. Em 2025, tecnologias que antes eram futuristas, como Inteligência Artificial e interfaces de voz, são parte integrante de muitas experiências digitais. Além disso, há uma consciência cada vez maior da necessidade de construir produtos para todos. Nossas histórias devem evoluir para abraçar essas novas realidades, garantindo que nossos produtos sejam modernos, inclusivos e relevantes.

Inteligência Artificial

Como seria uma história que incorpora **Inteligência Artificial**? Em vez de o usuário realizar toda a ação, ele pode se beneficiar de um sistema proativo. Exemplo: "Como um ouvinte frequente do nosso app de música, eu quero receber uma playlist personalizada chamada 'Descobertas da Semana' toda segunda-feira, para que eu possa descobrir novos artistas com base no meu gosto musical sem esforço."

Interfaces de Voz (VUI)

Para **Interfaces de Voz (VUI)**, o contexto é rei. A história precisa refletir a situação do usuário, que muitas vezes é de "mãos e olhos ocupados". Exemplo: "Quando estou cozinhando e com as mãos sujas, eu quero pedir ao meu assistente de cozinha para 'adicionar dois ovos à minha lista de compras', para que eu não me esqueça do item sem precisar tocar em nenhuma tela".

Acessibilidade

Já a **acessibilidade** não pode ser uma reflexão tardia; ela deve ser tecida em nossas histórias desde o início, tratando pessoas com deficiência como personas de primeira classe. Exemplo: "Como um usuário cego que utiliza um leitor de tela, eu quero que todos os botões da interface sejam corretamente rotulados, para que eu possa navegar pelo aplicativo de forma autônoma e eficiente", em conformidade com as diretrizes da WCAG.

Histórias com Consciência: Design Sustentável e Ético

A nossa responsabilidade como criadores de produtos digitais vai além da funcionalidade e da usabilidade. As decisões que tomamos têm um impacto real no mundo — no meio ambiente, através do consumo de energia dos data centers, e na sociedade, através da forma como lidamos com os dados e a atenção dos nossos usuários. Podemos e devemos usar as User Stories como uma ferramenta para embutir considerações éticas e de sustentabilidade diretamente no nosso processo de desenvolvimento.

🌱 Design Sustentável

O **Design Sustentável** no mundo digital foca em criar serviços que sejam eficientes e tenham o menor impacto ambiental possível. Isso pode ser traduzido em histórias muito práticas.

🛡️ Design Ético

No campo do **Design Ético**, as histórias podem nos ajudar a proteger o usuário e a construir confiança. Em vez de usar "dark patterns" para enganar as pessoas, podemos escrever histórias que promovam a transparência.

🌍 Exemplo de História Sustentável

"Como um usuário consciente do consumo de dados, eu quero que as imagens do site sejam carregadas em formatos otimizados e com o tamanho adequado para a minha tela, para que a página carregue mais rápido, consuma menos bateria e reduza a pegada de carbono da minha navegação".

🔒 Exemplo de História Ética

"Como um novo usuário, eu quero ver um resumo claro e em linguagem simples de como meus dados serão utilizados antes de criar minha conta, para que eu possa tomar uma decisão informada sobre minha privacidade".

Por exemplo: "Como um usuário consciente do consumo de dados, eu quero que as imagens do site sejam carregadas em formatos otimizados e com o tamanho adequado para a minha tela, para que a página carregue mais rápido, consuma menos bateria e reduza a pegada de carbono da minha navegação". Esta história liga diretamente a performance à sustentabilidade.

No campo do **Design Ético**, as histórias podem nos ajudar a proteger o usuário e a construir confiança. Em vez de usar "dark patterns" para enganar as pessoas, podemos escrever histórias que promovam a transparência. Por exemplo: "Como um novo usuário, eu quero ver um resumo claro e em linguagem simples de como meus dados serão utilizados antes de criar minha conta, para que eu possa tomar uma decisão informada sobre minha privacidade". Ao colocar essas histórias no backlog e priorizá-las, transformamos nossos valores em funcionalidades concretas e demonstramos um compromisso genuíno com o bem-estar do nosso usuário e da sociedade.

A Magia nos Detalhes: Histórias para Microinterações e Feedback

As grandes funcionalidades de um produto são como os capítulos de um livro — elas movem a trama principal para frente. Mas a qualidade da prosa, as frases bem construídas e as pequenas descrições que dão vida ao cenário são o que realmente nos encantam. No mundo digital, essas "frases bem construídas" são as **microinterações**. São as pequenas animações, os sons sutis e as respostas visuais que confirmam nossas ações e tornam a experiência mais fluida e humana.

Gatilho

Ação do usuário que inicia a microinteração

Regras

O que acontece quando o gatilho é ativado

Feedback

Resposta visual, sonora ou tátil para o usuário

Loops e Modos

Condições que determinam se a microinteração se repete

Uma microinteração é um momento contido em um produto que realiza uma única tarefa e, crucialmente, fornece feedback ao usuário. Pense na leve vibração e na animação de um coração quando você curte uma postagem, ou na mudança de cor de um botão quando você passa o mouse sobre ele. Esses detalhes parecem pequenos, mas têm um impacto gigantesco na percepção de qualidade e na usabilidade. Eles comunicam estado, confirmam ações e previnem erros, tudo isso enquanto adicionam um toque de personalidade à interface.

> ✨ Exemplo de História para Microinteração:
>
> "Como usuário, ao arrastar um item para a minha lista de tarefas concluídas, eu quero ver uma animação suave e ouvir um som de 'check' sutil, para que eu tenha um feedback satisfatório de que a tarefa foi completada".

Para garantir que esses detalhes encantadores não se percam no meio do desenvolvimento de grandes funcionalidades, eles merecem suas próprias User Stories. Por exemplo: "Como usuário, ao arrastar um item para a minha lista de tarefas concluídas, eu quero ver uma animação suave e ouvir um som de 'check' sutil, para que eu tenha um feedback satisfatório de que a tarefa foi completada". Os critérios de aceitação seriam específicos sobre a duração da animação e as características do som. Escrever histórias para microinterações é a marca de equipes maduras que entendem que uma ótima experiência do usuário é a soma de muitos pequenos momentos bem projetados.

A Arte da Escolha: Uma Introdução à Priorização

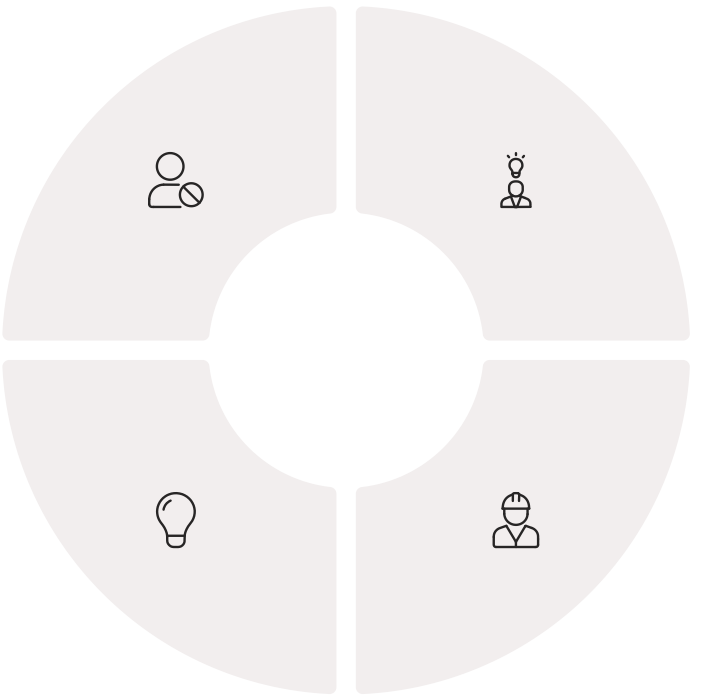
Um Product Backlog bem nutrido é um sinal de uma equipe cheia de ideias e com um profundo entendimento do seu usuário. No entanto, ele também apresenta um desafio constante: com tantas boas histórias para escolher, o que devemos construir a seguir? Se tentarmos fazer tudo de uma vez, não faremos nada bem. A priorização é, portanto, uma das habilidades mais críticas na gestão de produtos. É a arte e a ciência de tomar decisões difíceis sobre onde focar a energia limitada da nossa equipe.

Valor para o Usuário

Resolve uma dor real? Melhora significativamente a experiência?

Valor para o Negócio

Gera receita ou crescimento? Reduz custos operacionais?



Risco/Aprendizado

O que aprendemos ao construir isso? Qual o risco de falha?

Esforço para Construir

É rápido e barato ou lento e caro? Qual a complexidade técnica?

O erro comum é pensar na priorização como um evento único. Na realidade, é um processo contínuo de negociação e reavaliação. O que era a maior prioridade na semana passada pode ser ofuscado por um novo aprendizado ou uma mudança no mercado hoje. Uma equipe sem um método claro para priorizar corre o risco de ser guiada pela "opinião mais alta" na sala ou de trabalhar em funcionalidades de baixo impacto simplesmente porque são fáceis de fazer.

Para navegar nesta complexidade, usamos frameworks que nos ajudam a pensar de forma estruturada. Embora a próxima aula seja inteiramente dedicada a este tópico, é vital entender o conceito agora. A priorização geralmente envolve pesar quatro fatores principais: o **valor para o usuário** (Resolve uma dor real?), o **valor para o negócio** (Gera receita ou crescimento?), o **esforço para construir** (É rápido e barato ou lento e caro?) e o **risco/aprendizado** (O que aprendemos ao construir isso?). Modelos como MoSCoW (Must-have, Should-have) ou RICE (Reach, Impact, Confidence, Effort) nos dão uma linguagem comum para debater essas escolhas. As histórias que escrevemos são as peças que movemos neste grande tabuleiro estratégico, sempre com o objetivo de maximizar o valor entregue.

Oficina Prática: Sujando as Mãos com Histórias

A teoria nos dá o mapa, mas só aprendemos a dirigir de verdade quando assumimos o volante. Chegou a hora de aplicar tudo o que discutimos em um cenário prático. Vamos exercitar nossa capacidade de enxergar o mundo através das lentes de Job Stories e User Stories.

🎯 Nosso Cenário

Estamos projetando um aplicativo móvel para ajudar estudantes universitários, nosso público-alvo, a encontrar e se candidatar a vagas de estágio e horas complementares. O objetivo é diminuir a ansiedade e a sobrecarga de informação que eles sentem ao procurar por essas oportunidades em diversos portais e murais da faculdade.

Exercício 1: Descobrindo a Motivação (Job Story)

Pense no contexto de um estudante universitário no meio do semestre. Qual é o gatilho, a situação que o faz pensar "eu preciso encontrar uma forma de ganhar experiência ou cumprir minhas horas"? Com base nisso, escreva uma Job Story completa.

Exercício 2: Traduzindo em Ação (User Story + Critérios de Aceitação)

Agora, vamos pegar uma parte dessa necessidade e transformá-la em uma User Story acionável para o nosso aplicativo. Foque na ação de buscar uma oportunidade. Depois, detalhe-a com pelo menos dois Critérios de Aceitação.

(Pense por um momento antes de ver o exemplo...)

> ### 📝 Exemplo de Resposta - Job Story:
>
> "**Quando** eu percebo que o final do semestre está se aproximando e ainda não cumpri minhas horas complementares obrigatórias, **eu quero** encontrar rapidamente oportunidades validadas pela minha universidade, **para que** eu possa garantir minha formatura sem comprometer meu tempo de estudo para as provas finais."

(Tente escrever a sua versão antes de continuar...)

> ### 📝 Exemplo de Resposta - User Story + ACs:
>
> **User Story:** "Como um estudante universitário, eu quero filtrar as oportunidades por área de interesse e carga horária, para que eu possa encontrar rapidamente vagas que se encaixem no meu curso e na minha disponibilidade."
>
> **Critério de Aceitação 1: Dado** que estou na tela de busca, **quando** eu seleciono o filtro "Design" e "10 horas semanais", **então** a lista de resultados é atualizada para mostrar apenas as vagas que correspondem a ambos os critérios.
>
> **Critério de Aceitação 2: Dado** que apliquei um filtro, **quando** eu clico no botão "Limpar Filtros", **então** todos os filtros são removidos e a lista volta a exibir todas as oportunidades disponíveis.

Consolidação e Seus Próximos Passos

Nesta aula, fizemos uma jornada profunda do "porquê" para o "o quê". Partimos da constatação de que os produtos mais bem-sucedidos não nascem de uma lista de funcionalidades, mas de uma compreensão genuína do problema e da motivação humana — o "trabalho" que o usuário precisa que seja feito. Aprendemos a capturar essa motivação com as Job Stories e a traduzi-la em ações concretas e gerenciáveis com as User Stories. Vimos como refinar essas histórias com o padrão INVEST e como torná-las à prova de ambiguidade com Critérios de Aceitação claros, organizando tudo em um Product Backlog dinâmico.

1 Primeiro o Problema

Antes de esboçar qualquer solução, force-se a articular o problema do usuário em uma única frase clara.

2 Pense em "Quando"

Inicie suas conversas sobre necessidades do usuário com a pergunta "Quando isso acontece?". O contexto revelará a verdadeira motivação para uma Job Story.

3 Checklist INVEST

Mantenha o acrônimo INVEST em mente sempre que escrever ou avaliar uma User Story. Ele é seu guia rápido de qualidade.

4 Defina "Pronto"

Nenhuma User Story está completa sem seus Critérios de Aceitação. Pergunte sempre: "Como saberemos que isso está funcionando como esperado?".

Autoavaliação

1. Qual é a principal diferença de foco entre uma Job Story e uma User Story? A) A Job Story foca no "quem" e a User Story no "como". B) A Job Story foca na solução e a User Story no problema. C) A Job Story foca na motivação e no contexto (problema), enquanto a User Story foca na funcionalidade e na interação (solução). D) Ambas têm o mesmo foco, mas são usadas por equipes diferentes.
2. (Estilo Concurso) De acordo com o acrônimo INVEST, uma User Story que depende de outra para ser iniciada viola principalmente o critério de ser: A) Valuable (Valiosa) B) Estimable (Estimável) C) Small (Pequena) D) Independent (Independente)
3. Qual elemento é usado para remover a ambiguidade de uma User Story, definindo as condições de sucesso e falha? A) O Product Backlog B) Os Critérios de Aceitação (ACs) C) A Job Story D) O critério "Negotiable" do INVEST
4. A estrutura "Quando..., eu quero..., para que..." é característica de qual artefato? A) Critério de Aceitação B) User Story C) Épico D) Job Story
5. **Questão Discursiva:** Descreva brevemente por que uma equipe de produto que se concentra apenas em User Stories, sem nunca considerar os "Jobs to Be Done", pode correr o risco de construir um produto que ninguém quer usar.

Gabarito e Próximos Passos

1-C	2-D	3-B	4-D
Questão 1	**Questão 2**	**Questão 3**	**Questão 4**
Job Story foca no problema, User Story na solução	Viola o critério Independent (Independente)	Critérios de Aceitação (ACs)	Job Story

> ### 📝 Resposta Discursiva (Exemplo):
>
> Uma equipe focada apenas em User Stories pode se tornar muito eficiente em construir funcionalidades ("o quê"), mas sem a visão estratégica do "Jobs to Be Done" ("o porquê"), corre o risco de construir funcionalidades que não resolvem uma necessidade real e profunda do usuário. Ela pode acabar otimizando partes de uma solução para um problema que não é o mais importante, resultando em um produto funcionalmente correto, mas que falha em ser "contratado" pelos usuários.

Conexão com a Próxima Aula

Agora que temos um backlog repleto de histórias bem formuladas, o desafio se torna: por onde começar? Na **Aula 17 – Priorização de Funcionalidades e Requisitos**, mergulharemos em métodos e frameworks para organizar estrategicamente nosso backlog, garantindo que a equipe esteja sempre trabalhando no que gera o máximo de valor para o usuário e para o negócio.

📚 Recursos Adicionais

- **Livro "Intercom on Jobs-to-be-Done":** Uma leitura concisa e prática para aprofundar na teoria e aplicação do JTBD no dia a dia.
- **Artigo "User Stories with Examples and a Template" da Atlassian:** Um guia de referência excelente e direto ao ponto para consultar ao escrever suas histórias.

NOTA IMPORTANTE: As informações regulatórias/legais/técnicas desta aula estão atualizadas até 2025. Consulte sempre fontes oficiais para verificar alterações.