Testes Exploratórios

1. O que são Testes Exploratórios

Testes exploratórios são uma abordagem de teste em que o testador investiga o sistema **sem seguir casos de teste pré-definidos**, aprendendo sobre o software enquanto testa.
O objetivo é **descobrir falhas, comportamentos inesperados e riscos** por meio da exploração livre, usando a experiência, a curiosidade e o pensamento crítico do testador.
Nesse tipo de teste, **planejamento, execução e análise acontecem ao mesmo tempo**, simulando o uso real do sistema por um usuário.

2. Por que Testes Exploratórios são usado

A principal razão para usar os Testes Exploratórios é o seu benefício investigativo. Ao contrário da verificação passiva de roteiros pré-definidos, esta abordagem transforma o testador em um investigador ativo, onde o aprendizado, o design do teste e a execução ocorrem simultaneamente.

Agilidade e Feedback: Esta metodologia é particularmente adequada para cenários onde a velocidade é crítica. Quando uma equipe precisa aprender rapidamente sobre um novo produto ou fornecer feedback imediato sobre uma nova funcionalidade, a exploração supera a documentação pesada. Ela permite avaliar a qualidade do produto diretamente da perspectiva do usuário final, sem o filtro de especificações técnicas rígidas.

Em muitos ciclos de desenvolvimento de software, especialmente nas iterações iniciais (Early Iterations), as equipes frequentemente não dispõem de tempo hábil para estruturar casos de teste formais. Nestes cenários, o teste exploratório é extremamente útil, preenchendo a lacuna de qualidade instantaneamente enquanto o produto ainda está evoluindo.

Cobertura de Riscos: O valor estratégico da exploração aparece na descoberta do desconhecido. É uma técnica essencial para encontrar novos cenários de teste e ampliar a cobertura além do "caminho feliz".

- **Aplicações de Missão Crítica:** Ao testar sistemas onde falhas não são toleráveis, o teste exploratório garante que os edge cases não resultem em falhas críticas de qualidade.
- **Suporte ao Processo:** A exploração não descarta a documentação; ela a reorienta. As sessões exploratórias podem auxiliar na definição de processos de teste unitário e gerar

registros detalhados que servirão de base para testes extensivos (ou automação) em sprints futuros.

Estrutura: Embora baseados na liberdade criativa, os testes exploratórios não são caóticos. Para garantir que o "Porquê" seja atingido com eficiência, utiliza-se o ciclo de Gerenciamento de Teste Baseado em Sessão (SBTM), ilustrado no diagrama que você compartilhou. Este ciclo garante foco e rastreabilidade através de quatro pilares:

1. **Planejar (CARTA DE PROJETO):** O teste começa com uma missão clara. A "Carta de Projeto" (Charter) guia o testador, definindo o que investigar sem ditar como fazer passo a passo.
2. **Rastrear (GERENCIAMENTO BASEADO EM SESSÃO):** O foco é mantido através de sessões cronometradas (*time-boxes*), garantindo que o tempo seja gasto produtivamente nas áreas de maior risco.
3. **Capturar (ANOTAÇÃO):** Durante a execução, o testador registra ideias, dúvidas e comportamentos anômalos. O registro ocorre em paralelo à investigação.
4. **Compartilhar (*DEBRIEF*):** O ciclo se fecha com uma conversa curta (*Debrief*) para inspirar a equipe, discutir descobertas e transformar o aprendizado individual em conhecimento coletivo.

Mas, é importante ressaltar que, apenas em casos raríssimos, o teste exploratório será a única forma de testar um software. No geral, para garantir a qualidade total, é necessário juntar vários modelos (como testes automatizados e scripts de regressão) em uma estratégia híbrida e robusta.

3. Principais Técnicas de Testes Exploratórios

4. Exemplos de Uso e Limitações

Exemplos de uso: os testes exploratórios são muito usados em sistemas novos, quando ainda não existem casos de teste definidos. Eles ajudam a explorar o sistema e identificar falhas que não foram previstas inicialmente.

Também são aplicados em funcionalidades críticas, como login, checkout em e-commerce e transferências bancárias, onde qualquer erro pode impactar diretamente o usuário. Além disso, funcionam como um complemento aos testes automatizados, ajudando a encontrar problemas inesperados. Em ambientes ágeis, como *squads* com sprints curtos, esse tipo de teste é bastante comum.

Exemplo prático: um exemplo simples é o fluxo de cadastro de usuário, feito sem roteiro fixo. O testador preenche os campos de diferentes formas, usando nome muito longo, caracteres

especiais ou e-mail inválido.

Esse tipo de exploração ajuda a encontrar falhas de validação e comportamentos que não aparecem nos testes automatizados.

Exemplos Práticos – Cenários:

- **Teste de Integração:** nesse cenário, o objetivo é identificar problemas de comunicação entre módulos. O testador executa funcionalidades diferentes ao mesmo tempo e observa como elas se comportam juntas.
- **Teste de Estresse**
  Aqui o foco é avaliar a estabilidade do sistema. O teste acontece simulando muitos usuários acessando a aplicação ao mesmo tempo, observando falhas, lentidão ou travamentos.

Limitações e Desafios: Os testes exploratórios podem exigir mais tempo, dependendo do sistema. Eles também dependem bastante da experiência do testador, o que pode fazer falhas passarem despercebidas.

Outro ponto é a dificuldade de documentação e repetição, já que nem sempre é fácil refazer exatamente o mesmo caminho testado. Além disso, não existe uma métrica clara de cobertura. Por isso, os testes exploratórios não substituem testes formais, principalmente em sistemas com requisitos críticos.