Testing I - Mesa de Trabalho - 05/11/2021

**Grupo 2**

* Anna Karla Américo
* Guilherme Chehade
* Mario Braga
* Thiago Maurat Martins Dias
* Vinícius Alexandre

 Exercício de Fixação

# Objetivo

* Compreender os processos de testes estáticos e processos de revisão.
* Familiarizar-se com o conceito de Documento de Requisito.
* Começar a relacionar os conceitos vistos nas aulas anteriores e suas consequências com os conceitos atuais.

# Micro desafios - Etapa I

Em sua mesa de trabalho, elabore uma tabela comparativa entre os Testes Estáticos e os Testes Dinâmicos.

Além disso, você deve desenvolver um quadro conceitual com todos os conceitos relacionados até agora, incluindo:

* **Os 7 Princípios de Testing**

R: 1) Teste demonstra a presença de defeitos: Esse princípio descreve que os testes têm a capacidade de que erros desconhecidos permaneçam no sistema. Porém, apesar disso, mesmo que nenhum defeito seja encontrado, isto não enseja que o software esteja livre de defeitos.

2) Teste exaustivo é impossível: Apesar da automação de software abarcar diversas possibilidades, o número de possibilidades possíveis é gigantesco, ao passo que afirmar que foi efetuado todos os testes é considerado impossível.

3) Teste antecipado: Os testes devem ocorrer o mais rápido possível durante seu desenvolvimento, tendo um objetivo e planejamento.. Com isso, são diminuídos os custos com a produção, correção e tempo.

4) Agrupamento de defeitos: Este princípio descreve que 80% dos defeitos são causados por 20% do código. Assim, na ocasião que é identificado essa parte sujeita a falhas, é possível que ocorra sua priorização enquanto deve haver a procura de erros nas demais regiões.

5) Paradoxo do pesticida: Há a necessidade dos testes serem revisados com frequência. Caso haja uma padronização a ser sempre executada, existirá uma analogia com o paradoxo do pesticida, eis que em determinado momento eles deixam de ser úteis por não conseguir encontrar novos erros.

6) Teste é dependente do contexto: Cada aplicação possui seu contexto próprio e exige que seus testes sejam específicos. Assim, diante da necessidade de exigir a aplicação de diferentes testes, não é possível executar os mesmos testes de um aplicativo bancária em um e-commerce.

7) A ilusão da ausência de defeitos: Por último, este princípio menciona que a ausência de defeitos não significa que não há erros a serem descobertos. Porém, só isso não é suficiente, visto que a isonomia de seu sistema não será equilibrada caso não seja encontrado qualquer tipo de bug e, mesmo assim, seu software não atender às expectativas do usuário.

* **Os 3 pilares do desenvolvimento de software**

Resposta: Ao contrário do que muitos podem supor, o processo de produção de um software em nada se parece com aquele linear de uma linha de montagem industrial. Questões como clientes indecisos, requisitos mal especificados, prazos apertados, linhas de código mágicas e testes “vagalumes” não são novidade para nenhum programador experiente, e medidas que se adaptem a demandas específicas de cada projeto são mais do que necessárias.   
*A qualidade funcional* → Consiste na eficiência do código, no quão bem a aplicação cumpre o seu papel. Esse aspecto é extremamente importante, principalmente para os usuários, que são os maiores afetados pela performance e confiabilidade do sistema.  
*Qualidade estrutural* → É a qualidade do código escrito e o quão bem estruturado ele é. Pontos como modularização e re-usabilidade são essenciais nesse aspecto para garantir que o código é de fácil compreensão e manutenível.   
*Qualidade de processo* → consiste na adoção de técnicas e estratégias que tornem o processo de implementação do produto o mais equilibrado possível..

* **Níveis de Teste**

Resposta:

**Teste de componente:** Os objetivos do teste de componente são: Reduzir o risco; Verificar se os comportamentos funcionais e não funcionais do componente estão de acordo com os projetados e especificados; Gerar confiança na qualidade do componente; Encontrar defeitos no componente; Prevenir a propagação de defeitos para níveis de teste superiores.

**Teste de Integração:** O teste de integração se concentra nas interações entre componentes ou sistemas. Os bejtivos são: Reduzir o risco; Verificar se os comportamentos funcionais e não funcionais das interfaces estão de acordo com os projetados e especificados; Gerar confiança na qualidade das interfaces; Encontrar defeitos — que podem estar nas próprias interfaces ou dentro dos componentes ou sistemas; Prevenir a propagação de defeitos para níveis de teste superiores.

**Teste de Sistema:** Os objetivos são: Reduzir o risco; Verificar se os comportamentos funcionais e não funcionais do

sistema estão de acordo com os projetados e especificados; Validar se o sistema está completo e funcionará conforme o esperado; Gerar confiança na qualidade do sistema considerando o como um todo;Encontrar defeitos; Prevenir a propagação de defeitos para níveis de teste superiores ou para produção.

**Teste de Aceitação:** O teste de aceitação, como o teste de sistema, geralmente se concentra no comportamento e nos recursos de todo um sistema ou produto. Os objetivos do teste de aceitação incluem: Estabelecer confiança na qualidade do sistema como um todo; Validar se o sistema está completo e funcionará conforme o esperado; Verificar se os comportamentos funcionais e não funcionais do sistema são os especificados.

Os níveis de teste se caracterizam pelos seguintes atributos:

-Objetivos específicos

-Bases de teste

-Objeto de teste

-Defeitos efalhas característicos

-Foco e responsabilidades específicas

* **Tipos de Testes**

Resposta:  
\*1. Teste funcional  
2. Teste regressivo  
3. Teste exploratório  
4. Teste de segurança  
5. Teste de performance  
\*6. Teste de integração  
\*7. Teste unitário  
8. Teste de usabilidade  
\*9. Teste de aceitação  
10. Teste de acessibilidade

* **Técnicas de Testes**

**Resposta:**

* Análise de valor limite: Testar um valor imediatamente abaixo do que o limite, o valor limite e um valor imediatamente acima do valor limite.
* Particionamento de equivalência: Esta técnica é usada para reduzir o número de casos de teste a um nível gerenciável, mantendo ainda uma cobertura razoável do teste.
* Tabela de decisão: Basicamente a Tabela de Decisão consiste em analisar as combinações entre as condições de um teste e definir qual o resultado esperado.
* **Erros, Defeitos e Falhas**

**Resposta:** Um erro é um ato falho do profissional que cria o código, um defeito é um problema existente em seu código, a falha é percebida quando o código é executado.

* **Casos de Teste**

**Resposta:** Documento com as informações sobre o que testar, os processos do teste e o resultado esperado.

Todo caso de teste deverá conter: Identificador, nome, descrição, precondição, passos e o resultado.

* Relatório de Defeitos

**Resposta:** Nesse relatório, deve-se listar os bugs encontrados, para que possam ser isolados e corrigidos. Assim como nos casos de teste, os bugs precisam ter seu identificador e os passos para reproduzi-lo. As informações deste relatório devem ser bem específicas para serem corrigidas.

Não deixe de discutir/dialogar com os colegas da mesa de trabalho.

**Bom trabalho!**