**Teste estático**

* O que é?

Teste estático é uma técnica de verificação do código, design e documentação de um software sem executá-lo. É realizado, normalmente, nas fases iniciais do desenvolvimento do software, antes que ele esteja pronto para testes.

O teste estático pode ajudar a encontrar erros, inconsistências e defeitos no design, especificações e requisitos do software. Ele também pode melhorar a segurança, a facilidade de manutenção e a legibilidade do código.

O teste estático pode ser realizado de forma manual ou com o auxílio de ferramentas, como: Analisadores de código, Revisões de código, Instruções passo a passo, Inspeções, Auditorias.

O teste estático é o oposto de “teste dinâmico”, que verifica o software executando o código. Com o teste estático, é possível encontrar erros em um estágio inicial de desenvolvimento, pois é mais fácil identificar a causa do erro na fonte.

O teste estático avalia entregas como:

* Módulos do programa
* Documento de Requisitos de Negócios
* Casos de uso
* Sistema e os requisitos funcionais
* Design
* Protótipo
* Documento de Especificação do Protótipo
* Planilha de dicionário do banco de dados
* Casos de teste a serem usados
* Manual do usuário
* Exemplos de testes estáticos incluem:

**Linting check:** Análise estática do código que verifica se as regras de estilo estão sendo seguidas. Por exemplo, o ESLint é uma ferramenta de linting que verifica se as declarações var estão sendo substituídas por let ou const.

**Análise estática:** Examina todos os caminhos de execução possíveis e seus valores variáveis. É uma forma de revelar erros que podem ficar ocultos por semanas, meses ou anos após o lançamento.

**FindBugs:** Analisador estático que procura erros no código Java

**Teste unitário**

* Oque é?

Um teste unitário é uma técnica de teste de software que verifica se as unidades individuais de um código estão funcionando corretamente.

Os testes unitários são realizados em unidades isoladas de código, como métodos ou funções, para verificar se o código se comporta de acordo com o esperado. Eles são automatizados e devem ser rápidos e baratos de executar.

Os testes unitários são fundamentais para economizar tempo e esforço, e são uma das técnicas mais importantes na pirâmide de testes. Eles ajudam a: Reduzir erros, Garantir a confiabilidade do sistema, Economizar tempo nos testes de regressão, Receber alertas quando um comportamento existente é quebrado.

Para escrever testes unitários, é possível utilizar frameworks de testes unitários e teste unitário runners.

Algumas boas práticas para escrever testes unitários são:

* Dar nomes aos métodos de teste que ajudem a entender os requisitos do código
* Escolher um modelo de teste e manter-se fiel a ele
* Garantir que um teste só tenha sucesso se o código que ele testa estiver correto
* Criar mensagens curtas e significativas de falha
* Exemplos de testes unitários incluem:
* **JUnit 5**: Uma ferramenta de teste unitário
* **Jest**: Uma ferramenta de teste unitário
* **pytest**: Uma ferramenta de teste unitário
* **PHPUnit**: Uma estrutura de teste para PHP
* **Mocha**: Uma estrutura de teste para Javascript
* **RSpec**: Uma estrutura de teste para Ruby
* **NUnit**: Um quadro de teste unitário para todas as línguas .NET
* **QUnit**: Uma ferramenta de teste unitário para JavaScript

**Teste de integração**

* Oque é?

Testes de integração são uma etapa do processo de desenvolvimento de software que verifica se os componentes de um sistema interagem corretamente, eles são realizados após os testes unitários e antes dos testes de sistema:

* Verificam se os módulos ou serviços do aplicativo funcionam bem juntos
* Garantem que a comunicação entre sistemas é eficiente e segura
* Identificam e corrigem problemas de interface
* Validam a interação entre diferentes componentes, como bancos de dados, APIs, segurança, entre outros

Os testes de integração são importantes para garantir a qualidade do software e evitar falhas e conflitos no sistema. Eles podem ser realizados de forma manual ou automatizada e são geralmente feitos em fases, começando pelos módulos mais simples.

Os testes de integração são diferentes dos testes unitários, que verificam se os elementos individuais do sistema estão corretos. Os testes de integração, por outro lado, verificam se a interação entre esses elementos ocorre da forma planejada.

Os testes de integração podem ser classificados em duas categorias: vertical e horizontal:

* O teste vertical verifica a interação entre os componentes de um mesmo módulo ou camada
* O teste horizontal testa a interação entre diferentes módulos ou camadas do sistema.
* Exemplos de testes de integração:
* Verificar se o cadastro de um usuário está funcionando corretamente quando integrado com o envio de e-mails de confirmação
* Testar a integração entre o serviço de pagamento e o banco de dados
* Testar a integração entre o serviço de envio de e-mails e a interface web
* Testar a integração entre o serviço de autenticação e a interface web

**Teste de integração**

* Oque é?

Testes end-to-end (E2E) são uma metodologia de teste de software que avalia a funcionalidade de uma aplicação do início ao fim, simulando a interação de um usuário:

* Verificam se todas as partes de um aplicativo estão funcionando conforme o esperado
* Identificam falhas que outros testes podem não detectar
* Garantem que o sistema funcione como esperado do início ao fim
* Validam todo o fluxo da aplicação

Os testes E2E são cruciais no desenvolvimento de software e reforçam a confiabilidade do software.

Para garantir a eficácia dos testes E2E, é importante:

* Isolar o ambiente de testes para evitar interferências
* Utilizar dados de teste representativos
* Configurar uma rede consistente
* Integrar serviços externos e APIs no ambiente de testes
* Implementar soluções de monitoramento e acompanhamento de logs

Algumas ferramentas para automatizar testes E2E são: Selenium, Playwright e Cypress.

* Exemplos de testes E2E:
* Visitar a página de login
* Preencher dados
* Clicar em botões
* Validar mensagens de sucesso
* Abrir o site de uma loja online
* Pesquisar produtos
* Adicionar itens ao carrinho
* Concluir o pagamento
* Verificar o recebimento do pedido