Testes Automatizados

13/08/2025

Na Aula de Hoje

- O que testar?
- Tipos de testes: funcionalidade, desempenho e segurança
- Estratégias e exemplos
- Discussão e atividade prática

O que testar?

- **Funcionalidades centrais**
  - Autenticação (login/logout)
  - CRUD (criação, atualização, deleção)
  - Processos essenciais ao negócio

O que testar?

- **Elementos reutilizados**
  - Middlewares
  - Cálculos recorrentes (ex.: média de notas)
  - Services usados em vários pontos do sistema

O que testar?

- **Casos conhecidos de problemas**
  - Você pode criar testes para os problemas que estão ocorrendo atualmente, e tentar arruma-los para fazer o teste passar.
  - Criar teste para garantir que o erro não volte a ocorrer

Evitando regressões

- **Testes como rede de segurança**
  - Uma mudança não deve quebrar algo que já funcionava
  - Sempre que um bug for corrigido → criar teste para ele
  - Testes ajudam a manter a confiança no código

Níveis de Teste

1. **Testes Unitários**
   - Funções e métodos isolados

2. **Testes de Integração**
   - Módulos trabalhando juntos
3. **Testes de Sistema**
   - Fluxo completo como o usuário vê
4. **Testes de Aceitação**
   - Confirmam se o software atende aos requisitos do cliente

---

Do menor para o maior

- Começar **unitário** → mais rápido, barato e específico
- Evoluir para **integração** → garantir que peças se encaixam
- Finalizar com **sistema/aceitação** → visão real de uso

Tipos de Teste

- **Funcionalidade**
  - Valida se o sistema faz o que foi especificado
- **Desempenho**
  - Mede tempo de resposta e carga suportada
- **Segurança**
  - Verifica autenticação, autorização e proteção de dados

Ex. : Funcionalidade

- **Login**:
  - Senha correta → acesso liberado
  - Senha errada → acesso negado
  - Conta bloqueada → mensagem de bloqueio

Ex.: Desempenho

- Enviar 1.000 requisições simultâneas → medir tempo médio de resposta
- Avaliar consumo de memória em processos pesados
- Monitorar comportamento sob alta carga

Segurança - Exemplo

- Tentativa de SQL Injection no login
- Verificar se senha é armazenada com hash seguro
- Testar rotas privadas sem autenticação

Ferramentas Úteis

- **Unitário**: Jest, Mocha, PHPUnit, PyTest
- **Integração**: Cypress, Playwright, Selenium

- **Desempenho**: JMeter, k6
- **Segurança**: OWASP ZAP, Burp Suite

Boas Práticas

- Nome claro e objetivo para cada teste
- Um teste = um cenário específico
- Evitar dependência entre testes
- Usar dados de teste representativos

Perguntas para Discussão

- Quais funcionalidades são mais críticas no seu projeto?
- Quais erros passados poderiam voltar a ocorrer?
- Onde o desempenho é mais importante no sistema?
- Quais riscos de segurança seu projeto pode enfrentar?

Atividade Prática

1. Escolha **2 funcionalidades centrais** do seu projeto
2. Identifique **cenários de teste** para elas
3. Defina pelo menos:
   - 2 teste de funcionalidade
   - 1 teste de integração
   - 1 teste de desempenho (opicional)
   - 1 teste de segurança
4. Escreva os casos de teste (não precisa codificar agora, só idealizar)