Poder Executivo
Ministério da Educação
Universidade Federal do Amazonas
Instituto de Ciências Exatas e Tecnologia

UFAM

Trabalho Prático 3

Teste de Sobrecarga

Versão 1.0

Data de Entrega:
27/11/2024

Integrantes:
Arnald Lucas Bentes
José Santo de Moura Neto
Nivaldo Yenar Cidade Rego

1. Conceito do Teste de Carga

O **teste de carga** é um tipo de teste de desempenho que avalia como um sistema, aplicação ou infraestrutura se comporta quando submetido a um volume de trabalho esperado ou maior que o normal.

Objetivo

- **Verificar Resiliência**: Avaliar se o sistema mantém o desempenho adequado com uma quantidade específica de usuários ou operações.
- **Identificar Limites**: Descobrir o ponto máximo em que o sistema começa a apresentar falhas ou degradação.
- **Prevenir Problemas em Produção**: Simular cenários reais para antecipar possíveis falhas em ambientes controlados.

Onde é mais eficaz

- Sistemas que esperam grandes picos de usuários, como e-commerces durante promoções ou redes sociais em eventos globais.
- Aplicações críticas como serviços bancários ou plataformas de saúde, onde a estabilidade é crucial.
- APIs e micro serviços que precisam lidar com várias requisições simultâneas.

Quando deve ser utilizado no ciclo de desenvolvimento

- **Após o desenvolvimento de funcionalidades críticas**: Para verificar como elas escalam.
- **Antes do lançamento em produção**: Para validar se a infraestrutura suporta o tráfego esperado.
- **Em atualizações de sistemas**: Após alterações que podem impactar o desempenho, como troca de servidores ou otimizações no código.

2. Vantagens e Limitações do Teste de Carga

Vantagens

1. **Prevenção de Falhas**: Identifica pontos críticos antes que o sistema entre em produção, reduzindo o risco de interrupções.

2. **Capacidade de Escala**: Ajuda a entender o limite máximo de usuários ou processos que o sistema pode suportar.
3. **Otimização de Recursos**: Permite ajustar a infraestrutura (servidores, rede, etc.) para atender à demanda sem excesso ou desperdício.
4. **Simulação de Cenários Reais**: Avalia o desempenho em situações próximas à realidade, como picos de acesso.
5. **Contribuição para a Qualidade**: Garante que o sistema ofereça respostas rápidas e estáveis mesmo em condições extremas.

Limitações

1. **Custo Elevado**: Ferramentas, configuração de ambientes de teste e recursos podem ser caros.
2. **Demora no Setup**: Planejar e configurar um teste realista pode consumir tempo significativo.
3. **Resultados Limitados**: Simulações podem não capturar todos os aspectos de um ambiente real, como comportamento humano imprevisível.
4. **Necessidade de Especialização**: Requer conhecimento técnico para configurar testes eficazes e interpretar os resultados.
5. **Dependência do Ambiente**: Se o ambiente de teste não for fiel ao de produção, os resultados podem ser menos confiáveis.

3. Contexto de Aplicação do Teste de Carga

O teste de carga é mais eficaz em cenários onde o desempenho do sistema é crítico para atender a demandas específicas. Os principais contextos incluem:

- **Plataformas de e-commerce**: Garantir que o site suporte picos de acessos, como durante promoções (exemplo: Black Friday).
- **Sistemas bancários e financeiros**: Assegurar estabilidade em operações como fechamento de lotes ou pagamentos simultâneos.
- **Aplicações corporativas**: Verificar a performance de sistemas internos usados por um grande número de colaboradores.
- **Sistemas de streaming**: Avaliar como a plataforma lida com um número elevado de transmissões simultâneas.
- **APIs e serviços em nuvem**: Validar a capacidade de resposta de APIs diante de uma alta taxa de requisições.

O teste deve ser aplicado durante as fases de desenvolvimento ou pré-produção, antes de grandes lançamentos ou atualizações. Ele é especialmente útil em sistemas com crescimento de usuários previsto ou em serviços onde interrupções causam grandes prejuízos financeiros ou de reputação.

O **Cypress** não é comumente utilizado para testes de carga (que geralmente requerem ferramentas específicas como JMeter ou Locust), mas pode ser adaptado para simular cenários de múltiplos acessos concorrentes em casos básicos. Aqui está como você pode usar o Cypress para esse propósito:

Para testar um cenário simples de carga usando **Cypress**, você pode criar um script que simula múltiplas visitas à sua aplicação. Embora o Cypress não seja projetado para realizar testes de carga em larga escala, ele pode ser útil para simular um número básico de acessos, foi feito um código para testar o comportamento da aplicação com várias requisições simultâneas.

Codigo utilizado para o teste:

```
describe('Simulação de Carga Simples com Cypress', () => {
  it('Simula múltiplos acessos à aplicação', () => {
    const numRequests = 50; // Número de requisições simuladas

    // Loop para simular múltiplos acessos
    for (let i = 0; i < numRequests; i++) {
      cy.visit('https://seusite.com') // Substitua pela URL da aplicação
        .then(() => {
```

```
        // Verifica se a página foi carregada corretamente
        cy.title().should('include', 'Título esperado da página');
      });
    }
  });
});
```

Limitações do Cypress para Teste de Carga

- **Não é uma ferramenta de carga real**: Cypress não é projetado para gerar alto tráfego de usuários simultâneos. Ferramentas como JMeter ou Gatling são mais adequadas para testes de carga real.
- **Recursos limitados**: O Cypress executa as requisições uma por vez e na mesma máquina, o que pode não ser representativo do uso simultâneo de múltiplos usuários em grande escala.

Conclusão

Embora o Cypress não seja ideal para testes de carga em grande escala, ele pode ser útil para testar a resposta da aplicação a várias requisições de maneira básica. Para testes mais robustos, considere usar ferramentas como **JMeter** ou **Locust**.