




Automação de testes com Playwright para o Sauce Demo

Gislane F. Nunes



Objetivo: Automatizar 10+ cenários de teste críticos.

Ferramentas:

- **Playwright:** Escolhido por sua velocidade, capacidade multi-browser e robustez.
- **JavaScript:** Linguagem nativa e ágil para o ecossistema web.

Arquitetura: Page Object Model (POM)

- Código Limpo e Organizado
- Evita Repetição
- Facilita a Manutenção Futura



Cenários

12 Cenários Implementados:

- ✓ Login com Credenciais Válidas
- ✓ Login com Credenciais Inválidas
- ✓ Login com Usuário Bloqueado
- ✓ Adicionar Item ao Carrinho
- ✓ Remover Item do Carrinho **(pela Página de Inventário)**
- ✓ Remover Item pela Página do Carrinho
- ✓ Ordenação de Produtos por Preço
- ✓ Finalizar a Compra com Sucesso (Fluxo End-to-End)
- ✓ Checkout sem Preencher Dados
- ✓ Cancelar a Compra na Tela de Overview
- ✓ Persistência do Carrinho Após Logout
- ✓ Integridade Visual das Imagens dos Produtos

Estratégia de Testes

(Topo) Nível 3: Testes de Análise Crítica (Edge Cases & Resiliência)

- **Objetivo:** Ir além do "caminho feliz". Pensar em cenários que, embora menos comuns, impactam a confiança e a experiência do usuário. É aqui que demonstramos uma análise crítica da aplicação.

(Meio) Nível 2: Testes da Jornada do Usuário (End-to-End)

- **Objetivo:** Simular o fluxo completo e mais comum que um usuário realizaria. O foco é garantir que a principal proposta de valor do e-commerce funcione de ponta a ponta.

(Base) Nível 1: Testes de Fundação (Smoke Tests)

- **Objetivo:** Garantir que as funcionalidades mais críticas e básicas da aplicação estão operando. Sem elas, o resto do sistema não pode ser testado.





Demonstração

```
npx playwright test tests/checkout.spec.js --headed
```

Execução de um dos fluxos mais importantes, o de checkout: a automação vai logar, selecionar um produto, navegar até o carrinho, preencher as informações do usuário e finalizar a compra.



Principais Desafios

Problema: Testes instáveis ("flaky") que falhavam apenas no Firefox em execução paralela.

Solução: Refatoração dos seletores para IDs dinâmicos e isolamento da execução do Firefox, garantindo 100% de estabilidade.



Próximos Passos:

- Integração com pipeline de CI/CD (GitHub Actions).
- Expandir para testes de API e testes visuais.



Repositório

<https://github.com/gf-nunes/desafio-voidr-playwright>



OBRIGADA!

Linkedin: <https://www.linkedin.com/in/gislanefnunes/>

GitHub: <https://github.com/gf-nunes>