

# Título: Desafio Prático de QA (Full-Stack)

**Objetivo:** Este desafio foi desenhado para avaliar um conjunto completo de habilidades de QA, incluindo:

1. A sua mentalidade analítica e capacidade de realizar testes exploratórios.
2. A sua habilidade técnica em diagnosticar problemas de front-end (DevTools).
3. A sua capacidade de testar e validar serviços de back-end (APIs).

O desafio é dividido em duas partes.

---

## Parte A: Teste Web Exploratório e Técnico (Swag Labs)

**Foco:** Análise de interface (UI), experiência do usuário (UX) e diagnóstico técnico (DevTools).

### Ambiente de Teste:

- **Site:** <https://www.saucedemo.com>
- **Contexto:** Este é um site de demonstração de um e-commerce.

**Credenciais de Acesso:** A senha para todos os usuários é: `secret_sauce`

1. **Usuário Padrão (Baseline):** `standard_user`
  - Use este usuário primeiro para entender o "caminho feliz" (o fluxo que *deveria* funcionar perfeitamente).
2. **Usuário com Problemas (Foco do Teste):** `problem_user`
  - Este usuário foi configurado para encontrar inconsistências e bugs na plataforma.

## Sua Missão (Parte A)

1. **Reconhecimento:**
  - Faça login com o `standard_user` e execute o fluxo principal de compra (adicionar item, ir ao carrinho, fazer checkout e finalizar). O objetivo é apenas entender como o sistema *deveria* se comportar.
2. **Teste Exploratório:**
  - Faça logout e entre com o `problem_user`.
  - Explore o site livremente, focando no fluxo de compra, mas também em outras áreas (ordenação, detalhes de produtos, etc.).

- Identifique e anote os bugs ou comportamentos inesperados que encontrar.

## O Que Entregar (Parte A)

Selecione os **2 (dois) bugs que considera mais críticos** e reporte-os no formato abaixo.

**Formato do Reporte (para cada um dos 2 bugs):**

- **Título:** (Breve e direto. Ex: "Erro de Cálculo no Total do Carrinho")
- **Severidade:** (Sua avaliação: Crítico, Alto, Médio ou Baixo)
- **Ambiente:** (Ex: Usuário: `problem_user`, Navegador: Chrome, OS: Windows)
- **Passos para Reproduzir (PpR):**
  1. (Passo 1. Ex: "Fazer login com 'problem\_user'...")
  2. (Passo 2. Ex: "Adicionar o item 'Sauce Labs Backpack'...")
  3. (Passo 3. Ex: "Clicar no ícone do carrinho...")
- **Resultado Esperado:** (O que *deveria* ter acontecido)
- **Resultado Atual:** (O que *realmente* aconteceu)

★ **Requisito Técnico Adicional (Apenas para o Bug #1 - O mais crítico):** Juntamente com o reporte do bug de maior severidade, anexe uma **evidência técnica** que ajude o desenvolvedor a diagnosticar o problema. Pode ser:

- Um print da aba "**Console**" do DevTools mostrando um erro de JavaScript.
- Um print da aba "**Network**" mostrando uma chamada de API que falhou (ex: status 404 ou 500).
- Uma captura de tela da aba "**Elements**" mostrando um problema claro de HTML ou CSS.

---

## Parte B: Desafio Técnico de API (Restful-booker)

**Foco:** Teste de serviços (back-end) e compreensão do protocolo HTTP.

**Ambiente de Teste:**

- **Documentação da API:** <https://restful-booker.herokuapp.com/apidoc/index.html>
- **Contexto:** Esta é uma API demo para agendamento de hotéis.

## Sua Missão (Parte B)

Use uma ferramenta de sua preferência (Postman, Insomnia, etc.) para realizar as seguintes tarefas e validar o fluxo básico da API:

#### 1. Criar um Token ( `POST /auth` ):

- Gere um token de autenticação. As credenciais padrão estão na documentação (são `admin / password123`).

#### 2. Criar um Agendamento ( `POST /booking` ):

- Crie um novo agendamento com dados fictícios (JSON) à sua escolha.

#### 3. Validar o Agendamento ( `GET /booking/{id}` ):

- Use o ID do agendamento que acabou de criar na etapa 2.
- Faça uma chamada `GET` para buscar este agendamento e confirmar que os dados foram salvos corretamente.

#### 4. Escrever Testes (Assertions):

- Na sua ferramenta de API, adicione pelo menos **duas** validações automáticas (testes/assertions) em qualquer uma das chamadas acima.
- *Exemplos de validações:* Verificar se o Status Code da criação foi `200` , ou verificar se o `firstname` no `GET` é o mesmo que enviou no `POST` .

## O Que Entregar (Parte B)

Existem duas formas de entregar esta parte (escolha a que preferir):

- **Opção 1 (Preferencial):** Exporte a sua coleção do Postman/Insomnia (em formato `.json`) e anexe-a.
- **Opção 2:** Envie capturas de tela claras de *cada uma* das 3 requisições (Auth, POST, GET), mostrando a URL, o Body (se houver), a Resposta (Response) e, principalmente, a aba de **Testes (Tests/Assertions)** que criou.

---

## Submissão Final

Por favor, envie a sua resposta contendo:

1. Os 2 reportes de bug da **Parte A** (incluindo a evidência técnica do bug #1).
2. A coleção exportada ou os prints da **Parte B**.

Boa sorte!