



# Teste Técnico para Software Developer Engineer in Test (SDET)

---

## Introdução

---

Como parte do processo de contratação para a posição de SDET na voidr, este teste técnico avalia suas habilidades em automação de testes para a aplicação Reddit (<https://www.reddit.com/>), incluindo testes de interface (UI) no site e testes de API usando a Reddit API (<https://www.reddit.com/dev/api/>). O foco principal é sua capacidade de planejar estrategicamente testes com base em um contexto de negócio, além de implementar uma suíte de testes robusta usando ferramentas modernas.

## Objetivo

---

Demonstrar sua capacidade de:

- Criar um plano de testes estratégico que priorize cenários críticos com base no contexto de negócio do Reddit.
- Implementar testes automatizados para UI e API, garantindo alta qualidade e cobertura.
- Integrar testes em ambientes containerizados e pipelines de CI/CD.

## Ferramentas e Tecnologias

---

- **Playwright** para testes de UI (web) e API
- **Docker** para containerização
- **GitHub Actions** para integração contínua (CI)

## Pré-requisitos

---

- Crie uma conta no Reddit para testar fluxos autenticados (API).
- Para testes de API, crie uma app do tipo script e obtenha credenciais de autenticação (OAuth2) conforme descrito em <https://github.com/reddit-archive/reddit/wiki/OAuth2-Quick-Start-Example>.

- Configure um ambiente com Node.js, Playwright e Docker.

## Contexto de Negócio

---

O Reddit é uma plataforma social onde usuários criam, compartilham e interagem com conteúdo em comunidades chamadas subreddits. A missão do Reddit é conectar pessoas por meio de interesses compartilhados, oferecendo uma experiência confiável e fluida. Como uma empresa de tecnologia, o Reddit prioriza:

- **Engajamento do usuário:** Garantir que funcionalidades como login, navegação em subreddits, upvotes, comentários e pesquisa sejam rápidas e confiáveis, pois impactam diretamente a retenção de usuários.
- **Confiabilidade da API:** A API do Reddit é usada por desenvolvedores externos e pelo próprio site para recuperar posts, gerenciar votos e enviar comentários. Falhas na API podem interromper integrações e a experiência do usuário.
- **Escalabilidade:** O Reddit lida com milhões de usuários diários, exigindo que o site e a API sejam robustos sob alta carga.
- **Segurança:** Fluxos autenticados (e.g., login, votação) devem proteger os dados do usuário e prevenir acessos não autorizados.

### Dica para Planejamento de Testes:

Use o contexto de negócio para identificar cenários críticos. Por exemplo:

- Priorize testes para fluxos de alto impacto, como pesquisa, acesso a subreddits, interação com posts, que afetam diretamente o engajamento.
- Inclua cenários de erro para garantir tolerância a falhas.
- Considere a frequência de uso (e.g., pesquisa é usada por 90% dos usuários) e o risco de falha (e.g., bugs na API de votação podem afetar a confiança na plataforma).

## Teste Técnico

---

### 1. Planejamento de Testes

- Crie um plano de testes detalhado com base no contexto de negócio, incluindo:
  - **Cenários priorizados:** Liste os cenários de teste para UI e API, justificando sua importância (e.g., impacto no engajamento, risco de falha).
  - **Estratégias de cobertura:** Explique quais funcionalidades serão testadas e por quê (e.g., fluxos autenticados, cenários de erro).
  - **Sugestões adicionais:** Proponha testes complementares, como testes de desempenho ou monitoramento sintético, alinhados às prioridades do Reddit.

### 2. Testes de UI com Playwright

- Configure um projeto Playwright para testar o site do Reddit (<https://www.reddit.com/>).
- Organize os testes usando o padrão **Page Object Model**.

### 3. Testes de API

- Use a Reddit API (<https://www.reddit.com/dev/api/>) para escrever testes automatizados.

### 4. (Bônus) Integração com Docker

- Crie um **Dockerfile** que configure o ambiente para executar os testes de UI ou API.
- Garanta que os testes possam ser executados dentro do contêiner Docker.

### 5. (Bônus) Integração Contínua

- Configure um fluxo de trabalho no **GitHub Actions** para executar os testes automaticamente em paralelo a cada push.
- O fluxo deve executar os testes de UI e API.

## Entregáveis

---

- Um repositório **PRIVADO** no GitHub contendo todo o código, incluindo scripts de teste, Dockerfiles e configurações de CI.
- Estruture um vídeo apresentação de até 10 minutos contemplando:
  - O plano de testes e sua justificativa com base no contexto de negócio.
  - A cobertura dos testes para UI e API.
  - Problemas ou bugs encontrados durante os testes.
  - Oportunidade de evolução e próximos passos.

## Critérios de Avaliação

---

- **Planejamento de Testes:** A clareza e justificativa do plano de testes, com foco nos cenários mais críticos e alinhados ao impacto no negócio.
- **Cobertura e Precisão dos Testes:** A abrangência dos testes e a precisão na verificação das funcionalidades essenciais, incluindo cenários de erro e fluxos críticos.
- **Qualidade do Código:** A organização e a limpeza do código, com foco nas melhores práticas de automação de testes, como o uso do Page Object Model, e na manutenibilidade do código.
- **Integração e Uso de Ferramentas:** A utilização eficaz das ferramentas, como Playwright, Docker e GitHub Actions, para garantir automação e consistência nos testes.

- **Apresentação e Comunicação:** A clareza e objetividade na apresentação do vídeo, incluindo a explicação do plano de testes, as dificuldades encontradas, soluções aplicadas e sugestões de melhorias.

## Entrega

---

Envie o vídeo apresentação e o link para seu repositório **PRIVADO** GitHub para o email [victor.buchalla@voidr.co](mailto:victor.buchalla@voidr.co) com o título "SDET Voidr - Teste Técnico". Certifique-se de que o repositório é privado e que você concedeu acesso ao usuário <https://github.com/buchalla>.

## Dúvidas

---

Se tiver perguntas ou precisar de esclarecimentos, entre em contato conosco pelo [victor.buchalla@voidr.co](mailto:victor.buchalla@voidr.co).