Plano de Teste Completo para o LinkedIn

Este plano de teste abrangente visa garantir a qualidade do site do LinkedIn (https://www.linkedin.com/) em seus diversos aspectos.

1. Plano de Teste Funcional (PDF)

1.1. Casos de Teste

1.1.1. Navegação

* **Cenário Positivo:** Acessar a página inicial e navegar para diferentes seções, como "Home", "Network", "Jobs", "Messaging", e "My Network".
* **Cenário Negativo:** Tentar acessar uma página que não existe (e.g., "linkedin.com/invalid-page").

1.1.2. Busca

* **Cenário Positivo:** Pesquisar por um usuário, empresa ou vaga de emprego e verificar se os resultados são relevantes.
* **Cenário Negativo:** Pesquisar por um termo inválido ou inexistente e verificar se a mensagem de erro é adequada.

1.1.3. Login e Logout

* **Cenário Positivo:** Efetuar login com credenciais válidas e realizar logout com sucesso.
* **Cenário Negativo:** Tentar logar com credenciais inválidas e verificar a mensagem de erro.

1.1.4. Cadastro de Perfil

* **Cenário Positivo:** Criar um novo perfil com dados válidos e verificar se o perfil foi criado com sucesso.
* **Cenário Negativo:** Tentar criar um perfil com dados inválidos (e.g., email já existente, campos obrigatórios em branco) e verificar as mensagens de erro.

1.1.5. Submissão de Formulários

* **Cenário Positivo:** Submeter um formulário de contato com dados válidos e verificar se a mensagem de sucesso é exibida.
* **Cenário Negativo:** Tentar submeter um formulário com dados inválidos (e.g., campos obrigatórios em branco, formato de email inválido) e verificar as mensagens de erro.

1.2. Critérios de Aceitação

* Todos os links devem funcionar corretamente.
* Todos os campos obrigatórios devem ser validados.
* As mensagens de erro devem ser informativas e claras.
* A navegação entre as páginas deve ser suave e intuitiva.
* O tempo de carregamento das páginas deve ser rápido.
* O conteúdo deve estar livre de erros ortográficos e gramaticais.
* O design do site deve ser consistente e agradável.

1.3. Escopo do Teste

* Todas as páginas principais do LinkedIn serão testadas.
* Todos os principais fluxos de usuário serão testados, incluindo login, cadastro, busca, navegação, e interação com posts e perfis.
* As funcionalidades de mensagens, notificações e feed serão testadas.

1.4. Ambientes de Teste

* Navegadores: Chrome, Firefox, Safari, Edge.
* Dispositivos móveis: Android, iOS.
* Resoluções de tela: diferentes tamanhos de tela serão testados, incluindo desktop, tablet e mobile.

1.5. Riscos e Dependências

* Dependências: os testes podem ser afetados por mudanças no código-fonte do LinkedIn.
* Riscos: a descoberta de bugs pode atrasar o lançamento do site.

1.6. Métricas de Sucesso

* Taxa de defeitos: percentagem de casos de teste que falharam.
* Cobertura de teste: percentagem de funcionalidades cobertas pelos testes.
* Tempo de resposta: tempo médio para carregar as páginas.
* Taxa de erros: número de erros encontrados durante os testes.

2. Projeto de Automação em Cypress

2.1. Estrutura do Projeto

```
codigo_automacao/
% % %  cypress/
%    % % %  integration/
%    %    % % %  navigation.spec.js
%    %    % % %  search.spec.js
```

```
%   %   % % %  login.spec.js
%   %   % % %  signup.spec.js
%   %   % % %  profile.spec.js
%   %   % % %  forms.spec.js
%   % % %  support/
%   %   % % %  commands.js
%   %   % % %  plugins.js
%   % % %  fixtures/
%   %   % % %  user.json
%   % % %  plugins/
%       % % %  index.js
% % %  .env.development
% % %  .env.staging
% % %  .env.production
% % %  allure-results/
```

2.2. Código Cypress

navigation.spec.js

```javascript
describe('Navegação', () => {
  it('Navegar para a página inicial', () => {
    cy.visit('/');
    cy.title().should('eq', 'LinkedIn: Log in or sign up');
  });

  it('Navegar para a página "Network"', () => {
    cy.visit('/in');
    cy.get('.nav__button--primary').contains('Network').click();
    cy.url().should('include', '/in');
  });
});
```

2.3. Cobertura de Teste

* Casos de borda: testes com dados inválidos, erros e cenários inesperados.
* Casos de erro: testes que validam a exibição de mensagens de erro adequadas.
* Cenários com dados dinâmicos: testes que simulam diferentes cenários com dados variáveis, como nomes de usuários, empresas e cargos.

2.4. Integração com Allure

* Configurar o Cypress para gerar relatórios Allure usando o plugin `cypress-allure-plugin`.

* Adicionar o código necessário para gerar relatórios Allure nos scripts de teste Cypress.

2.5. Configurações de Variáveis de Ambiente

* Definir variáveis de ambiente específicas para cada ambiente (desenvolvimento, homologação, produção) no arquivo `.env` correspondente.
* Usar essas variáveis de ambiente nos testes Cypress para configurar URL, credenciais e outras configurações específicas de cada ambiente.

3. Instruções de Execução (README.md)

3.1. Configuração do Ambiente

1. Instalar Node.js e npm.
2. Instalar Cypress: `npm install cypress --save-dev`.
3. Criar um arquivo `.env` com as variáveis de ambiente.
4. Iniciar o Cypress: `npx cypress open`.

3.2. Execução dos Testes

* Executar todos os testes: `npx cypress run`.
* Executar um teste específico: `npx cypress run --spec cypress/integration/login.spec.js`.
* Executar testes em um ambiente específico: `npx cypress run --env environment=staging`.

3.3. Interpretação dos Resultados

* Os resultados dos testes serão exibidos no terminal e no dashboard do Cypress.
* Os relatórios Allure serão gerados na pasta `allure-results`.

3.4. Configuração de CI/CD

* Integrar os testes Cypress em um pipeline CI/CD, como GitHub Actions ou Jenkins, utilizando o Cypress CLI e a configuração de ambiente específica.

4. Plano de Testes de Performance (PDF)

4.1. Testes de Carga

* Usar ferramentas como JMeter ou k6 para simular um grande número de usuários acessando simultaneamente o LinkedIn.
* Monitorar o tempo de resposta, o throughput e a taxa de erros durante os testes de carga.

4.2. Testes de Estresse

* Aumentar gradualmente a carga no site até atingir o ponto de falha.
* Monitorar as métricas de performance e identificar os pontos críticos do sistema.

4.3. Testes de Capacidade

* Medir a capacidade máxima do site para lidar com um número específico de usuários e transações.
* Identificar os gargalos do sistema e melhorar o desempenho do site.

4.4. Métricas de Performance

* Tempo de resposta: tempo médio que o site leva para responder a uma solicitação.
* Throughput: número de solicitações processadas pelo site por segundo.
* Taxa de erros: percentagem de solicitações que falharam.

5. Plano de Testes de Segurança (PDF)

5.1. Testes de Vulnerabilidade

* Usar ferramentas de análise de vulnerabilidades como OWASP ZAP ou Burp Suite para identificar vulnerabilidades comuns, como injeção de SQL, XSS e CSRF.

5.2. Teste de Autenticação/Autorização

* Verificar se as políticas de autenticação e autorização estão implementadas corretamente, incluindo o tratamento de senhas e permissões de acesso.

5.3. Teste de Penetração

* Contratar uma empresa especializada em segurança para realizar testes de penetração e avaliar a resistência do site a ataques maliciosos.

6. Plano de Testes de Acessibilidade (PDF)

6.1. Conformidade com WCAG

* Testar o site contra as diretrizes WCAG (Web Content Accessibility Guidelines) para garantir a acessibilidade para todos os usuários, incluindo pessoas com deficiências.

6.2. Ferramentas de Acessibilidade

* Usar ferramentas como Axe, Lighthouse e NVDA para automatizar testes de acessibilidade e identificar problemas.

6.3. Casos de Teste Acessíveis

* Criar casos de teste específicos para validar a navegação com teclado, uso de leitores de tela, e acessibilidade para pessoas com deficiências visuais e motoras.

7. Estratégia de Teste de Regressão

7.1. Seleção de Casos de Teste

* Selecionar um conjunto de casos de teste crítico para garantir que as funcionalidades principais não sejam afetadas por novas alterações no site.

7.2. Automação de Regressão

* Automatizar os casos de teste de regressão para que eles possam ser executados automaticamente em cada build.

7.3. Planejamento de Execução

* Executar os testes de regressão em cada build, antes de cada lançamento, ou periodicamente, dependendo da frequência de alterações no código-fonte.

8. Documentação de Integração de APIs

* Se o site usa APIs, criar documentação de teste específica para as APIs, incluindo casos de teste, contratos e testes de carga.

8.1. Testes de API Automatizados

* Usar ferramentas como Postman ou Newman para automatizar testes de API e validar as respostas das APIs.

8.2. Testes de Contrato

* Usar ferramentas como Pact para definir os contratos das APIs e garantir que as APIs respeitam os contratos esperados.

8.3. Testes de Carga em APIs

* Usar ferramentas como k6 para simular um grande número de solicitações para as APIs e verificar o desempenho das APIs sob carga.

Organização de Arquivos e Pastas

* Todos os arquivos e pastas serão organizados de forma clara e acessível para facilitar a execução e manutenção dos testes.

* Os relatórios e artefatos serão armazenados em locais facilmente acessíveis para revisão posterior.

Conclusão

Este plano de teste completo fornece uma estrutura abrangente para testar o LinkedIn em seus diversos aspectos, incluindo funcionalidade, performance, segurança, acessibilidade e regressão.
A implementação deste plano garante a qualidade do site e a satisfação dos usuários.