Plano de Teste Funcional (PDF):

Casos de Teste:
1. **Login com Credenciais Corretas**:
   - **Cenário Positivo**: Usuário insere credenciais válidas e clica em "Login".
   - **Passos**:
     1. Acessar a página de login.
     2. Inserir um nome de usuário válido.
     3. Inserir uma senha válida.
     4. Clicar no botão "Login".
   - **Critérios de Aceitação**: Usuário é redirecionado para a página inicial após o login bem-sucedido.

2. **Login com Credenciais Incorretas**:
   - **Cenário Negativo**: Usuário insere credenciais inválidas e clica em "Login".
   - **Passos**:
     1. Acessar a página de login.
     2. Inserir um nome de usuário inválido.
     3. Inserir uma senha inválida.
     4. Clicar no botão "Login".
   - **Critérios de Aceitação**: Usuário recebe uma mensagem de erro informando que as credenciais são inválidas.

Escopo do Teste:
- Serão testadas as funcionalidades de login, navegação pelo feed, busca de usuários e publicações, e interações com postagens.

Ambientes de Teste:
- Navegadores: Chrome, Firefox, Safari.
- Dispositivos móveis: iOS e Android.
- Resoluções de tela: 1920x1080, 1366x768, 375x667.

Riscos e Dependências:
- Risco de instabilidade da conexão com a Internet.
- Dependência de APIs externas para carregamento de conteúdo.

Métricas de Sucesso:
- Taxa de defeitos: Menos de 5% de falhas nos testes.
- Cobertura de teste: 95% das funcionalidades testadas.
- Tempo médio de execução dos testes: Menos de 10 minutos.

Projeto de Automação em Cypress:

Estrutura do Projeto:

- `codigo_automacao/`
  - `testes/`
  - `fixtures/`
  - `comandos_personalizados/`
  - `plugins/`

Código Cypress:
```javascript
describe('Testes de Login', () => {
  it('Login com Credenciais Corretas', () => {
    // Implementação dos passos de teste
  });

  it('Login com Credenciais Incorretas', () => {
    // Implementação dos passos de teste
  });
});
```

Cobertura de Teste:
- Serão implementados testes para cobrir casos de borda, erro, e cenários com dados dinâmicos.

Integração com Allure:
- Configuração para geração de relatórios avançados de teste com o Allure.

Configuração de Variáveis de Ambiente:
- Variáveis de ambiente serão configuradas para suportar diferentes ambientes de teste.

Instruções de Execução (README.md):

Configuração do Ambiente:
1. Instale o Node.js.
2. Instale o Cypress.
3. Clone o repositório do projeto.

Execução dos Testes:
- Local: `npx cypress run`
- CI/CD: `npx cypress run --env environment=staging`

Interpretação dos Resultados:
- Resultados podem ser vistos no terminal e no dashboard do Cypress.
- Relatórios completos disponíveis no Allure.

Configuração de CI/CD:
- Integre os testes Cypress em sua pipeline utilizando GitHub Actions ou Jenkins.

Plano de Testes de Performance (PDF):

Testes de Carga:
- Utilização de JMeter para simular usuários simultâneos acessando o site.

Testes de Estresse:
- Identificação do ponto de falha do site através de testes de estresse com k6.

Testes de Capacidade:
- Definição de testes para medir a capacidade máxima do site sem degradação significativa da performance.

Métricas de Performance:
- Acompanhamento do tempo de resposta, throughput, e taxa de erros durante os testes.

Plano de Testes de Segurança (PDF):

Testes de Vulnerabilidade:
- Identificação de vulnerabilidades comuns como injeção de SQL, XSS, e CSRF.

Teste de Autenticação/Autorização:
- Verificação das políticas de autenticação e autorização implementadas no site.

Teste de Penetração:
- Testes de penetração para avaliar a resistência do site a ataques maliciosos.

Plano de Testes de Acessibilidade (PDF):

Conformidade com WCAG:
- Testes para garantir a conformidade do site com as diretrizes WCAG.

Ferramentas de Acessibilidade:
- Recomendação de ferramentas como Axe, Lighthouse, e NVDA para testes de acessibilidade.

Casos de Teste Acessíveis:
- Inclusão de casos de teste para validar a acessibilidade para todos os usuários.

Estratégia de Teste de Regressão:

Seleção de Casos de Teste:
- Identificação dos casos de teste críticos a serem incluídos na execução de regressão.

Automação de Regressão:
- Criação de scripts de regressão automatizados para garantir a estabilidade das funcionalidades existentes.

Planejamento de Execução:
- Definição da frequência de execução dos testes de regressão para cada build.

Documentação de Integração de APIs:

Testes de API Automatizados:
- Criação de casos de teste para validar as respostas das APIs utilizando Postman ou Newman.

Testes de Contrato:
- Garantia de que as APIs respeitam os contratos esperados através de testes específicos.

Testes de Carga em APIs:
- Detalhe de testes de carga direcionados para as APIs utilizando ferramentas como k6.

Certifique-se de seguir o plano de teste detalhadamente e manter todos os artefatos organizados e acessíveis para facilitar a execução e revisão dos testes. Boa sorte na execução dos testes para a página do Instagram! Ø=Þ€Ø=Ý Ø=ÜÊ