UNIVERSIDADE ESTADUAL DO CEARÁ

CENTRO DE CIÊNCIAS E TECNOLOGIA

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JONNAS CHRISTIAN SOUSA DE PAIVA

IMPLEMENTAÇÃO DE UM MOTOR SELF-HEALING GENÉRICO PARA TESTES AUTOMATIZADOS

FORTALEZA – CEARÁ

2025

SUMÁRIO

1 INTRODUÇÃO

A automação de testes é uma prática fundamental para garantir a qualidade de sistemas em ciclos de desenvolvimento ágil, especialmente diante da adoção crescente de práticas como integração e entrega contínuas (CI/CD). No entanto, um desafio recorrente enfrentado por equipes de qualidade é a fragilidade dos testes automatizados frente a mudanças frequentes na estrutura das interfaces web. Alterações aparentemente simples no DOM (Document Object Model), como a troca de um `id`, uma classe ou o reposicionamento de elementos, podem tornar seletores obsoletos e causar falhas em testes que antes funcionavam corretamente.

Diante desse cenário, esta proposta apresenta o desenvolvimento de um motor de *self-healing* para testes automatizados, com foco na análise de alterações no DOM e na readequação automática de seletores por meio de armazenamento e reaproveitamento de dados em arquivos JSON. A proposta adota uma abordagem genérica e extensível, baseada em heurísticas e regras estruturais, sem a necessidade de técnicas complexas de inteligência artificial, visando entregar uma solução prática, explicável e de fácil integração com diferentes ferramentas de automação.

1.1 MOTIVAÇÃO

Além da fragilidade natural dos testes automatizados diante de mudanças no DOM, outro fator que motivou este trabalho é a complexidade envolvida na configuração de soluções existentes de *self-healing*, como o Healenium. Embora funcionais, tais soluções muitas vezes exigem integrações específicas, são limitadas a determinados frameworks (por exemplo, o Healenium não é compatível com Cypress) e não oferecem uma abordagem verdadeiramente genérica.

Ao buscar alternativas, percebe-se que há uma lacuna no desenvolvimento de mecanismos de autoajuste que sejam simples, flexíveis e aplicáveis a diferentes contextos de automação. A proposta deste trabalho surge como uma resposta a essa necessidade, visando oferecer um motor leve e independente de plataforma, que possa ser facilmente adaptado aos mais diversos ambientes de teste automatizado.

1.2 OBJETIVO GERAL

Desenvolver um motor de *self-healing* genérico para testes automatizados, capaz de identificar mudanças no DOM e corrigir seletores de forma inteligente, garantindo maior

estabilidade e eficiência nos testes.

1.3 OBJETIVOS ESPECÍFICOS

- Implementar um módulo para extração de elementos do DOM, focando nos atributos essenciais (`id`, `name`, e `class`);
- Criar um mecanismo de comparação entre versões do DOM antes e depois das mudanças;
- Desenvolver estratégias para readequação automática dos seletores de teste;
- Integrar a solução com frameworks populares de automação de testes (ex.: Selenium, Cypress, Robot Framework);
- Avaliar a eficácia do motor em cenários reais de automação.

2 FUNDAMENTAÇÃO TEÓRICA

A automação de testes de software desempenha um papel cada vez mais estratégico na garantia da qualidade em ciclos de desenvolvimento modernos, especialmente diante da adoção de metodologias ágeis e integração contínua. No entanto, conforme apontado por Battina (BATTINA, 2019), embora o avanço das ferramentas de automação tenha reduzido o esforço humano em muitas tarefas repetitivas, ainda existe um obstáculo significativo: a manutenção de testes automatizados. A autora aponta que uma das principais dificuldades enfrentadas pelas equipes de QA é o esforço dedicado à manutenção dos testes, que pode representar até 40% do tempo total de trabalho.

2.1 CONCEITO E MECANISMOS DE SELF-HEALING

Frameworks *self-healing* são estruturas capazes de detectar e corrigir automaticamente falhas de testes que ocorrem devido a mudanças no sistema sob teste. Saarathy et al. (SAARATHY; BATHRACHALAM; RAJENDRAN, 2024) apresentam uma proposta de arquitetura que inclui monitoramento contínuo da execução dos testes, detecção de falhas estruturais e mecanismos automáticos de correção. A ideia central é permitir que os testes se adaptem a mudanças em tempo de execução sem necessidade de intervenção manual imediata.

Embora a proposta original dos autores envolva o uso de inteligência artificial, os princípios fundamentais — como a reavaliação de seletores quebrados e a aplicação de ajustes dinâmicos — podem ser implementados com abordagens mais simples e genéricas, baseadas em heurísticas e análise estrutural do DOM.

2.2 IDENTIFICAÇÃO DE SELETORES E FRAGILIDADE DOS TESTES

Uma das principais causas de falhas em testes automatizados de interface é a quebra de seletores, ou seja, a incapacidade do script de localizar elementos modificados ou renomeados na interface. Isso ocorre com frequência em sistemas que sofrem constantes evoluções visuais, o que torna a automação tradicional frágil e de alta manutenção.

Ao identificar mudanças no DOM e adaptar os seletores de forma dinâmica, torna-se possível contornar essas quebras e promover maior resiliência dos testes. Essa adaptação pode ser feita com base em atributos como classes, nomes, hierarquia de elementos ou proximidade relativa no DOM.

2.3 ARMAZENAMENTO E APRENDIZADO CONTÍNUO COM JSON

A proposta deste TCC envolve a utilização de arquivos JSON como meio para armazenar seletores identificados e suas alternativas. A cada execução, seletores corrigidos podem ser salvos, atualizando uma base reutilizável que evolui com o tempo. Isso viabiliza um processo contínuo de refinamento sem dependência de modelos preditivos ou algoritmos complexos.

Essa estrutura também permite rastreabilidade e integração fácil com frameworks de automação populares, além de fornecer transparência sobre as modificações realizadas automaticamente.

2.4 SÍNTESE DA FUNDAMENTAÇÃO

Frente ao desafio recorrente da quebra de testes automatizados por mudanças na interface, surgem soluções com diferentes níveis de complexidade. Este trabalho adota uma abordagem pragmática e extensível baseada em comparação de DOM, adaptação de seletores e versionamento em arquivos JSON — alinhada a princípios de self-healing, porém sem a dependência direta de inteligência artificial.

3 TRABALHOS RELACIONADOS

Esta seção apresenta e analisa estudos que abordam soluções práticas ou revisões aplicadas no contexto de automação de testes com foco em autorreparação. A seguir, destacam-se trabalhos que servem de referência para comparação e inspiração na proposta deste TCC.

3.1 KHANKHOJE (2023)

Khankhoje (KHANKHOJE, 2023) oferece uma análise abrangente dos principais frameworks de automação com capacidades *self-healing*, como Healenium e Functionize. O autor categoriza as abordagens em baseadas em DOM, visão computacional e aprendizado de máquina. Apesar da diversidade de soluções, muitas são limitadas por dependências comerciais ou falta de adaptabilidade. Isso reforça a necessidade de soluções mais simples e genéricas, como a proposta deste TCC.

3.2 TAMRAPARANI E DALAL (2023)

O estudo de Tamraparani e Dalal (TAMRAPARANI; DALAL, 2023) traz uma aplicação prática da automação autorregenerativa em ambientes regulatórios financeiros, destacando a importância da auditabilidade e rastreabilidade nos testes. A proposta deste TCC se beneficia dessa visão, incorporando práticas de versionamento e persistência de seletores para ambientes exigentes.

3.3 SÍNTESE

Ambos os trabalhos evidenciam a relevância do tema e inspiram a proposta aqui apresentada. No entanto, observa-se uma lacuna na aplicação de comparações estruturais de DOM e armazenamento inteligente via JSON como base primária de um motor de self-healing, o que este TCC busca explorar de forma prática e genérica.

4 METODOLOGIA

O projeto será desenvolvido em Python, utilizando Selenium para extração do DOM e JSON para armazenar as versões analisadas. A metodologia adotada inclui:

1. Implementação do módulo de extração de DOM, focado nos atributos essenciais (`id`, `name`, e `class`);
2. Comparação de versões extraídas antes e depois de mudanças estruturais;
3. Geração de sugestões automáticas para correção de seletores;
4. Integração da solução com frameworks de automação de testes;
5. Testes com páginas de diferentes níveis de complexidade para validação da eficácia do motor.

5 CRONOGRAMA

O desenvolvimento do projeto será dividido nas seguintes etapas:

Mês/2025	**Atividade**
Março	Extração de DOM e armazenamento em JSON
Abril	Implementação da comparação entre versões do DOM
Maio	Desenvolvimento do mecanismo de readequação automática
Junho	Integração com frameworks de automação e testes em cenários reais
Julho	Refinamento do motor e defesa do TCC

Tabela 1 – Cronograma de desenvolvimento do projeto

6 CONCLUSÃO

A elaboração desta proposta permitiu identificar de forma clara a motivação, os objetivos e o escopo do trabalho a ser desenvolvido. A necessidade de tornar os testes automatizados mais resilientes frente a mudanças estruturais e visuais nas aplicações web motivou a criação de um motor de *self-healing* baseado em análise de DOM e gestão de seletores.

O projeto foi estruturado em etapas bem definidas, com foco na simplicidade, reutilização e viabilidade prática. A abordagem proposta evita o uso de soluções complexas baseadas em inteligência artificial, priorizando mecanismos heurísticos e armazenamento inteligente em JSON, o que facilita a integração com frameworks existentes.

A expectativa é que o desenvolvimento ao longo dos próximos meses permita validar a efetividade da solução em cenários reais, contribuindo com avanços na área de testes automatizados e oferecendo uma alternativa leve, extensível e de fácil manutenção para equipes de QA.

REFERÊNCIAS

BATTINA, D. S. Artificial intelligence in software test automation: A systematic literature review. **Journal of Emerging Technologies and Innovative Research (JETIR)**, v. 6, n. 12, p. 1329–1332, 2019. ISSN 2349-5162. Disponível em: <https://www.researchgate.net/publication/357032804>.

KHANKHOJE, R. Effortless test maintenance: A critical review of self-healing frameworks. **International Journal for Research in Applied Science and Engineering Technology**, v. 11, n. 10, 2023.

SAARATHY, S. C. P.; BATHRACHALAM, S.; RAJENDRAN, B. K. Self-healing test automation framework using ai and ml. **International Journal of Strategic Management**, v. 3, n. 3, p. 45–77, 2024.

TAMRAPARANI, V.; DALAL, A. Self generating and self healing test automation scripts using ai for automating regulatory and compliance functions in financial institutions. **Revista de Inteligencia Artificial en Medicina**, v. 14, n. 1, p. 784–794, 2023.