iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: X Month of publication: October 2025

DOI: https://doi.org/10.22214/ijraset.2025.74864

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

AI-Driven Self-Healing Automated UI Testing Framework with Visual Proof

Payal Chede[1], P.A.Tijare[2]
Sipna College of Engineering and Technology, Amravati, India

Abstract: Automateduser-interface (UI)testingisessentialfor continuoussoftware delivery, yetitremains fragile in the face of frequent front-end changes. Conventional automation frame- works (Selenium, Cypress, Playwright) depend on static locators and basic assertion primitives that are prone to breakage and produce flaky results. Recent research and commercial developments propose AI-enabled strategies to reduce fragility: self- healing locators, perceptual visual checks, and intelligent reporting.This review synthesizes literature from 2018– 2025, identifies key deficiencies in existing techniques, and articulates a unified architecture that integrates(a) hybrid AI-driven locator healing, (b) semantic Visual Proof Validation (VPV) inspired by Vision Transformer architectures, and (c) dual-format AI reporting with automated rerun orchestration.We present comparative analysis, identify research gaps, and discuss prototype results demonstrating improved healing accuracy and actionable test intelligence.
Keywords: Self-Healing Automation, Visual Proof Validation, Test Automation, Vision Trans- former, LocatorHealing, AI Reporting, Software Quality.

I. INTRODUCTION

Automated User Interface (UI) testing has emerged as a cornerstone of contemporary software development lifecycles, especially in agile and continuous integration/continuous deployment (CI/CD) environments. Its adoption enables rapidrelease cycles while safeguardingfunctional correctness anduser experience consistency.However, the robustness of automated test suites remains a persistent concern due to the fragility of element locators and the ever- evolvingcomplexity of modern front-end frameworks such as React, Angular, and Vue [1].

Empirical industry studies indicate that between 30–60% of test maintenance efforts are devoted to mitigating test flakiness and repairing broken locators.The principal causes of such instability include Document Object Model (DOM) refactoring, cascading style sheet (CSS) class renames, dynamic identifiers, lazy-loading mechanisms, and responsive design adaptations across multiple viewports [2],[3].

Conventional automation tools—including Selenium Web- Driver, Cypress, and Playwright—offer robust APIs for DOM manipulation and test execution but exhibit limited intrinsic resilience against dynamic UI transformations. Consequently, automation engineers frequently expend significant effort maintaining existing test scripts rather than extending test coverage, thereby

diminishing return on investment (ROI) and slowing feedback cycles in CI pipelines [4]. Simultaneously, parallel advances in computer vision and deep learning have opened new possibilities for visual reasoning within software testing. Recent transformer-based vision architectures, such as Vision Transformers (ViT), have demonstrated the abilitytoderive rich semantic representations of UI images, surpassing traditional pixel-level comparison metrics [5]. This evolution suggeststhepotential tocombine functionalself-healing—theadaptive correction ofbroken locators— withsemanticvisual verification, ensuringnot onlythat an element is programmaticallylocated but alsothat its visual and spatial integrity is preserved This review consolidates advancements in AI-assisted auto- mated UI testing, providing:

1) Asynthesisofmajorresearchdevelopmentsandcommercialinnovationsfrom2018to2025.
2) Acomparativeassessmentofrule-based,heuristic,andAI-drivenself-healingapproaches.
3) Anexplorationoftheemergingintegrationofvisualvalidationthroughsemanticmodeling.
4) Adiscussionofunresolvedresearchgapsandopportunitiesforfutureexploration[6]

II. LITERATURE REVIEW

The literature on automated UI testing and self-healing has diversified across several streams. Below wesummarize the most relevant contributions and trends, emphasizing developments between 2018 and 2025.

A. *Rule-based and Heuristic Repair*

Early works addressed locator brittleness with deterministic fallbacks: multiple stored selectors, ordered try-catch replacement strategies, and string-similarity heuristics (Levenshtein distance, longest common subsequence).While these approaches can correct minor identifier drift, they fail under structural DOM reorganization or semantic re- labeling [7].

B. *DeepLearningandEmbeddingMethods*

Deep neural network (DNN) approaches introduced representation learning for DOM nodes: embedding textual content, attributes, and small DOM subtrees via sequence encoders or graph embeddings.These embeddingssupported similarity search and ranking. Some works also introduced attention mechanisms to weigh contextual featuresdynamically. Nonetheless, deepmodelsraised questionsabout interpretabilityand production readinessdueto inference cost.

C. *VisualRegressionandSemanticValidation*

Traditional visual testing used pixel diffs or SSIM (Structural Similarity Index), which are sensitive to trivial rendering differences (antialiasing, font rendering). The advent of transformer- based vision models (ViT) and contrastive pretraining enabled semantic-level visual comparison.Works in 2023–2025 suggest that ViT- derived embeddings combined with lightweight anomaly detectors can detect layout and semantic inconsistencies more robustly than pixel-diff approaches. Integration of these methods into testing pipelines is nascent but promising.[8]

D. *LLMsforTestGenerationandExplanation*

Large language models (LLMs) have been explored for test generation, translating user stories or bug reports into test scripts. LLMs alsoshowpromisein explainingtest failures in human-readableterms. However, LLMs arenot yet a turnkeysolution for runtime healing and visual proofing; their outputsrequire grounding and verification [9], [10].

E. *SummaryofTrends*

Overall, the field has moved from brittlerule-based practices toward hybrid AI systems that seek balance between robustness, interpretability, and computational cost. The remaining open areasare: combining functional healing with semantic visual verification, establishing explain- able confidence metrics, and achieving production-grade scalability across platforms and view-ports [11].

III. RESEARCH GAPS

Despite notable progress in AI-driven automation and self- healing testing, several unresolved issues continue to hinder the attainment of stable, transparent, and generalizable frame- works.

A. *Under-IntegrationofSemanticVisualVerification*

While significant progress has been achieved in locator self-healing, the integration of semantic visual verification (VPV) with functional healing remains inadequate.Existing automation frameworks primarily rely onpixel-level com- parisons or structural similaritymeasures (SSIM), which are overlysensitive to superficial rendering differences such as antialiasing, dynamic shadows, or font smoothing.Consequently, trivial aesthetic variations may be falsely interpreted as regressions, whereas deeper semantic anomalies—for ex- ample, an icon swap, missing button affordance, or truncated label—often go undetected. Modern Vision Trans-formers (ViT) and contrastive pretraining models can capture high- level semantic meaning from UI screenshots, yet few studies have effectively incorporated these into runtime automation pipelines. Moreover, current tools rarely establish cross-modal validation betweenDOM- based healingandvisualverification,leavinga disconnect between what theframework findsand whattheuser actually sees.Developing unified models that reason jointly over the DOM tree and its rendered visual con- text— possibly through multimodal embeddings or attention- fusion networks—remains an open research challenge. Such integration would enable more human-like understanding of layout integrity, functional affordances, and stylistic coherence [12], [13].

B. *Absence ofContinuousLearningPipelines*

Most self-healing systems employ offline-trained ML models that remain static once deployed. However, user interfacesevolve continuouslythrough incrementalreleases, leadingtoconcept drift—a gradual deviation between the model’s learned distribution andthecurrent production reality.

With-out adaptiveretraining, locator healing accuracy decays over time, diminishing ROI. Few frameworks implement online or incremental learning where test outcomes dynamically feed back into model updates.Such systems could leverage active learning to query human validators when uncertainty is high and gradually refine model boundaries with minimal supervision. Yet, this approach raises several complexities: Data governance: determining which healed examples should be stored and how to prevent reinforcement of incorrect map- pings. Versioning androllback: maintaininghistorical snapshots of locator models for audit and debugging. Drift detection: identifying when UI modifications represent genuine evolution versus transient testnoise.Theliteraturelacksformalmethodologiesfor designingsafecontinuous-learningarchitecturesthat preserve both accuracyand traceability, which constitutes a keygap for industrial deployment [14].

C. Integration ofTestGeneration,RuntimeHealing,and VisualValidation

Recent studies highlight the promise of Large Language Models (LLMs) for test-case generation. Yet, these capabilities remain decoupled fromruntimeself-healingandvisual verification processes.Generatedteststypicallyuse staticselectors, offeringno feedback when locators drift or the UI changes.There is minimal research on closed-loop frameworks where LLM- generated test scripts continuously interact with a self- healing engine and a VPV module. Such integration couldform areinforcement cycle:the LLM proposes or refines test stepsbased onhealingoutcomes, while visual feedback provides grounding signals to prevent se- mantic drift. Developing this synergy requires advances in: Contextual grounding between language instructions and real UI elements. Error correction mechanisms that update both code and ML models. Multi-agent orchestration, where vision, DOM, and language agents communicate under a unified policy network. This remains a largely unexplored frontier that could fundamentally transform how automated testing pipelines evolve over time [15].

D. Explainability,Interpretability,andAuditability

As frameworks adopt opaque models, explainability be- comes a crucial barrier. Test engineers must be able to justify why a locator was healed.A formal provenance layer linking each healing action to historical examples or training data would allow systematic validation and rollback.

E. Cross-PlatformGeneralization

Existing systems are predominantly web-centric and perform poorly when ported to mobile or hybrid environments, or when adapting to responsive design changes across viewports. Future research must address multimodal generalization

F. ScalabilityandComputationalEfficiency

Although deep learningapproaches deliver superior accuracy, theyoften incur high inference latencyandresource costs, making them impractical for large-scale CI/CD pipelines that run thousands of tests per build. Most industrial systems thus revert to hybrid approaches or lightweight heuristics to maintain throughput.Current research seldom addresses resource- aware model design, distributed inference, or edge- optimized healing agents capable of scaling efficiently. Techniques such as model quantization, knowledge distillation, or adaptive model loading could reduce runtime overhead without compromising accuracy [18]. A systematic framework for balancing precision, interpretability, and computational cost remains missing in current literature.

G. Security,Privacy,andEthical Considerations

Self-healing automation increasingly involves the collection of DOM snapshots, screenshots, and usage telemetry from pro-duction environments.Thesedata streamsmayinadvertentlycapturesensitiveinformation (user credentials, personal identifiers, or proprietaryUI designs). Yet, fewstudies explicitlyaddress data anonymization, secure storage, or ethical handling within AI-based testing systems [19].Future frameworks must embed privacy-preserving mechanisms such as federated learning, differential privacy, or synthetic data generation to ensure compliance with data protection regulations while maintaining model fidelity. The ethical governance of autonomous decision-making in testing—especially when it modifies pro- duction environments—also warrants closer scrutiny.

H. PrioritizedResearchOpportunities

Addressing the above gaps offers fertile ground for innovation.Priority directions include Hybrid multimodal healing systems that unify DOM, vision, and textual cues. Online and active learning pipelines with human-in-the- loop supervision. Explainable and auditable AI mechanisms for transparent decision tracking.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

Cross-platform generalization leveraging do- main adaptation and transfer learning.Open- source datasets and standardized metrics forbenchmarking andreproducibility[20].Energyefficientandprivacy-preservingarchitecturessuitableforenterprise deployment. Closed- loop LLM-integrated frameworks connecting test generation, healing, and VPV.

By systematically addressing these areas, the next generation of automated UI testing frame- works can evolve toward self-adaptive, trustworthy, and human-comprehensible systems capable of sustaining the velocity demandedby modern software delivery pipelines.

IV. COMPARATIVEANALYSIS

To contextualize the proposed framework, we performed a comparative evaluation of representative automation solutionsacrossfivedimensions:adaptability, healing accuracy, semanticvisual validation,reportingintelligence,and operational cost impact.Table I summarizes key contrasts.

TABLEI. COMPARATIVESUMMARYOFEXISTINGAUTOMATIONSOLUTIONS

Tool	Adaptability	Healing Accuracy	Visual Validation	Reporting
Selenium	Low	0%	None	Basic logs
Cypress	Medium	10–20%	None	JSON reports
Katalon	Medium	40–50%	Pixel Diff	Static HTML
Testim.io	Medium	65–80%	Screenshot Diff	Interactive Dashboard
Mabl	High	70–85%	Region Level	Analytics Dashboard

The primaryobservation fromTable Iisthat traditional open-source tools (Selenium, Cy- press) lackinherent self- healing capabilities, leaving the burden of maintenance entirely on the engineer.Commercial tools (Testim.io, Mabl) integrate AI for both locator healing and visual checks, but often rely on proprietary, opaque models and simplistic visual metrics (pixel or region diff) rather than deep semantic validation. Our proposed framework is designed to bridge this gap by offering a hybrid, explainable core that couples AI-driven healing with semantic Visual Proof Validation (VPV).

V. PURPOSE AND OBJECTIVES

The purpose of this work is to consolidate theresearchtrajectory of AI-enabled UItest automation and to proposea unified framework that addresses the identified limitations. The objectives are:

1) DesignObjective:ProposeanarchitecturalblueprintthattightlycoupleshybridAI-basedlocatorhealing with semantic Visual Proof Validation and AI-driven reporting.
2) TechnicalObjective:Define thealgorithms(conceptualandmathematical)forhybridhealingscoringand dual- layer VPV that are robust, explainable, and computationally tractable.
3) EmpiricalObjective:Evaluatetheframework’simpactonhealingaccuracy,maintenancetime,and visual mismatch detection in representative dynamic web applications.

VI. FRAMEWORK FLOW

The framework flow diagram (Figure 1) illustrates a six-stage continuous loop for resilient UI testing. Since this figure spans both columns, it will be positioned by LaTeX at the top of the current or next page to ensure optimal layout.

ThecontinuousloopforresilientUItestingconsistsofthefollowingstages:

1) InputStage:Definetestscriptsanddata.
2) BrowserAutomation:Executetests using tools (e.g.,Selenium/Playwright).If a failureoccurs(elementnot found), it triggers the next stage.
3) HealingStage:TheAI-drivencoreself-correctsfailingselectors(elementlocators)toenablere-execution.
4) Visual Proof Stage: Capturesscreenshotsand performsvisual diffing (comparison)be-fore andafter healing to verify visual correctness and provide proof.
5) Reporting:Generatesdetailedtestlogsandhealingreports.

6) HealingRe-run:Immediatelyreteststhecorrectedsteptoverifythefix,closingtheloopforcontinuous automated resilience.

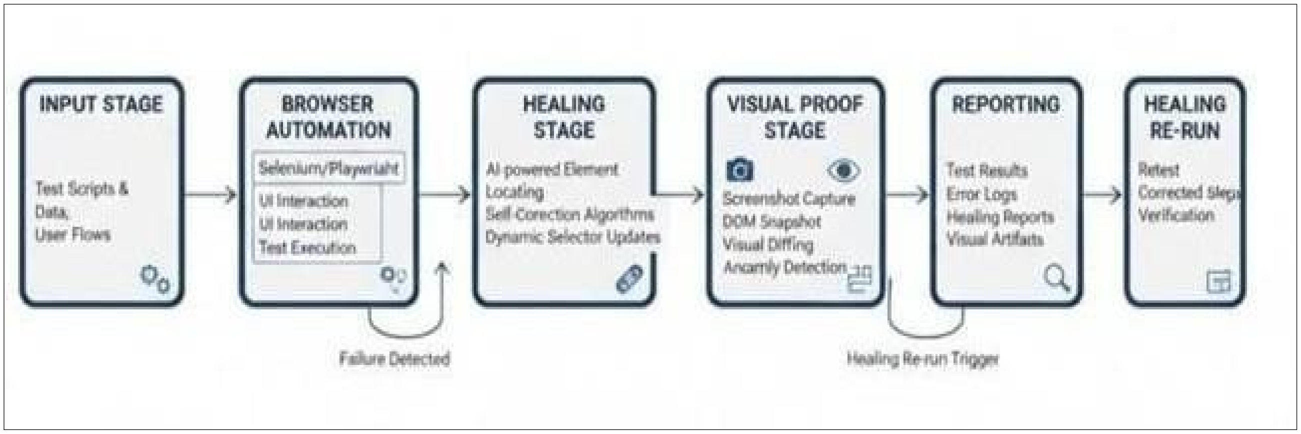

FIG.1.AI-DRIVENSELF-HEALINGUITESTINGFRAMEWORK:ARCHITECTUREFLOW

VII. CONCLUSION AND FUTUREWORKS

This review and prototype study argues that the next generation of UI test automation should transcend brittle locatorsandnaivevisualdiffsbyintegratinghybridAIhealing,semanticvisualproof,andactionableAIreporting.The combined approach yields substantial improvements in healing accuracy and maintenance efficiency while providing the explainability necessary for human trust and governance.

Future Directions

1) Full ViT Integration: Train and deployViT-based modelstuned for UI semantics (icon presence, alignment, text completeness).
2) Continuous Learning Pipeline: Buildarobust onlinelearninginfrastructurewith con-trolled feedback loopsand label curation
3) PublicBenchmarkSuite:ReleasedatasetsofmutatedDOMsandannotatedhealingoutcomestoenable comparativeresearch.
4) Cross-PlatformGeneralization:Extendthearchitecturetonativemobile,hybridapps,andmulti-browser environments.
5) GovernanceExplainability:Integrateattributionandprovenancereportingtosatisfyauditrequirementsin regulated domains.

REFERENCES

[1] D.S.Battina,“Artificialintelligenceinsoftwaretestautomation:Asystematicliteraturereview,”Int.J.Emerg.Technol.Innov.Res.,vol. 6, no. 5, pp. 120–127, 2019.
[2] R.Khankhoje,“AnIn-DepthReviewofTest AutomationFrameworks: TypesandTrade-offs,”IJARSCT,vol.3, no.1,pp.55–64,2023
[3] K.Pelluru,“AI-DrivenDevOpsOrchestrationinCloudEnvironments,”Int.J.Sci.Technol., vol.1,no.6, pp.1–15,2024.
[4] A.Jime´nez-Ram´ırezetal.,“Automatedtestinginroboticprocessautomationprojects,”J.Softw.:Evol.Process,vol.35,no.3,2023.
[5] Z.Liuetal.,“ChattingwithGPT-3for zero-shotmobileautomatedGUItesting,”arXivpreprintarXiv:2305.09434,2023.
[6] M.Scha¨fer etal.,“AnempiricalevaluationofLLMs forautomatedunittestgeneration,”IEEETrans.Softw.Eng.,2023.
[7] R.Feldtetal.,“Towardsautonomoustestingagentsvia conversationalLLMs,” inProc.ASEConf.,2023.
[8] K.Lietal.,“Utilizingdeeplearningtooptimizesoftwaredevelopmentprocesses,” arXivpreprintarXiv:2404.13630,2024.
[9] S.Lukasczyketal.,“Automatedunittestgeneration forPython,”EmpiricalSoftw.Eng.,vol.28,no.2,2023.
[10] Q.Guoetal.,“ExploringChatGPTinautomatedcoderefinement,”inProc.ICSE,2024.
[11] N.Alshahwanetal.,“AutomatedunittestimprovementusingLLMsatMeta,”inProc.FSE,2024.
[12] R.Patel andA.Saha,“Reviewonvisualregressiontestingframeworks,”ACMSIGSOFT Notes,vol.50,no.1,2025.
[13] Y.Chenetal.,“Transformer-basedvisual validationformobileUItesting,”SpringerLNCS,vol.14532,pp.310–322,2025.
[14] P.Singh,M.Gupta,andA.Mehra,“Adaptiveself-healinglocators withLLM-driventestrepair,”IEEE Access,vol.13,pp.56789–56803,A.2025.
[15] J.Zhang,R.Lin,andP.Bose,“IntegratingVisionTransformers withDOMcontextforUIanomalydetection,”IEEETrans.Neural
[16] Netw.Learn.Syst.,vol.36,no.9,pp.12045–12059,2025.
[17] M.H.Leeand A.Fernandez,“LLM-assistedself-healing frameworksfor end-to-endtesting,”EmpiricalSoftw.Eng.,vol.30,no.4,A.2025.
[18] V.SharmaandK.Rao,“Surveyonmultimodal learningfor softwarequalityassurance,”ACMComput.Surv.,vol.56,no.8,2024.
[19] C.D.Nguyen,“RobustUIlocatorrecoveryusingcontrastivepretraining,” Softw.Test.Verif.Reliab.,vol.34,no.1,2025.
[20] T.Oliveira etal.,“Self-healingtestcasesvia contextualattention mechanisms,”inProc.IEEEICSEWorkshops,2024.
[21] D.Patel,“EvaluatingexplainableAItechniquesforvisualvalidation,”IEEEAccess,vol. 2025