

The Automation Framework Trap

Why Banking QA Needs a Redesign

WHITE PAPER

CONTENTS

Executive Summary	03
The Framework Illusion: A Familiar Banking Story	04
The Real Cost of Framework-First QA	05
Manual Testers: The Untapped Asset	07
Enter AI + NLP: A New Model for Regulated Banking	08
Real-World Impact: Stories from Forward-Looking Banks	10
Implementation Blueprint	11
The Future of Banking QA	11
About Testsigma	12

Executive Summary



Banks have long relied on complex test automation frameworks to keep pace with digital transformation. Yet many of these efforts fail to deliver lasting impact. Despite well-staffed teams and custom tooling, **70% of defects in banking software are still found manually** often in UAT or even production¹. This is despite the fact that most banks already have automation pipelines in place.



88% of test automation frameworks fail to scale beyond a few power users², making automation a checkbox activity instead of a risk-reducing force.

This white paper explains why frameworks fail, outlines the hidden costs of current approaches, and introduces a modern, AI-powered testing model that empowers manual testers to automate with speed, accuracy, and context.

1 The Framework Illusion: A Familiar Banking Story

In many banks, QA teams have spent years building and refining automation frameworks using tools like Selenium and Playwright. Testing has gradually morphed into a framework-first discipline, where success is measured by tool mastery rather than actual risk coverage. But behind the scenes, adoption remains shallow.

The original essence of testing-deep domain knowledge, exploratory insight, and understanding of business logic - has been sidelined. Tools, once intended to accelerate and support testers, have taken center stage. The result is brittle, siloed systems maintained by a few technical specialists. They show passing pipelines while bugs go unnoticed.

At one of the largest banks in the world, a senior technology leader shared that despite investing in Playwright-based automation, the tool "never found the bugs" - manual testers did. This isn't an isolated case.

In just 26 months, nine major UK banks reported over 800 hours of IT outages, many caused by software bugs that passed automated pipelines but failed in production³.

It's a pattern across the industry

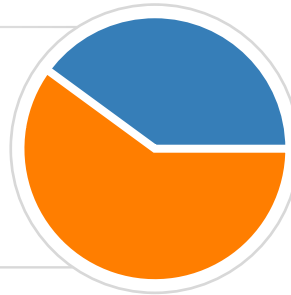
Frameworks validate the obvious, while real defects hide in complex, business-critical flows that only experienced testers can anticipate.



2 The Real Cost of Framework-First QA

Test automation frameworks come with a high maintenance burden. Compliance and business logic remain untested - not due to negligence, but because most of QA's time is consumed by maintaining brittle test suites. QA leaders take the blame for failures they didn't cause, while Playwright pipelines show green and dev-owned tests miss what matters.

QA engineers spend 40 – 60% of their time on test maintenance fixing flaky scripts and updating locators, rather than exploring new risk areas.



40–60% QA time consumed by test maintenance

Actual risk exploration

Regional banks across the world have seen the impact of weak automation strategies.

In the UK, for example, a major bank experienced a **mobile app outage** that blocked access for millions of customers following an internal software update.



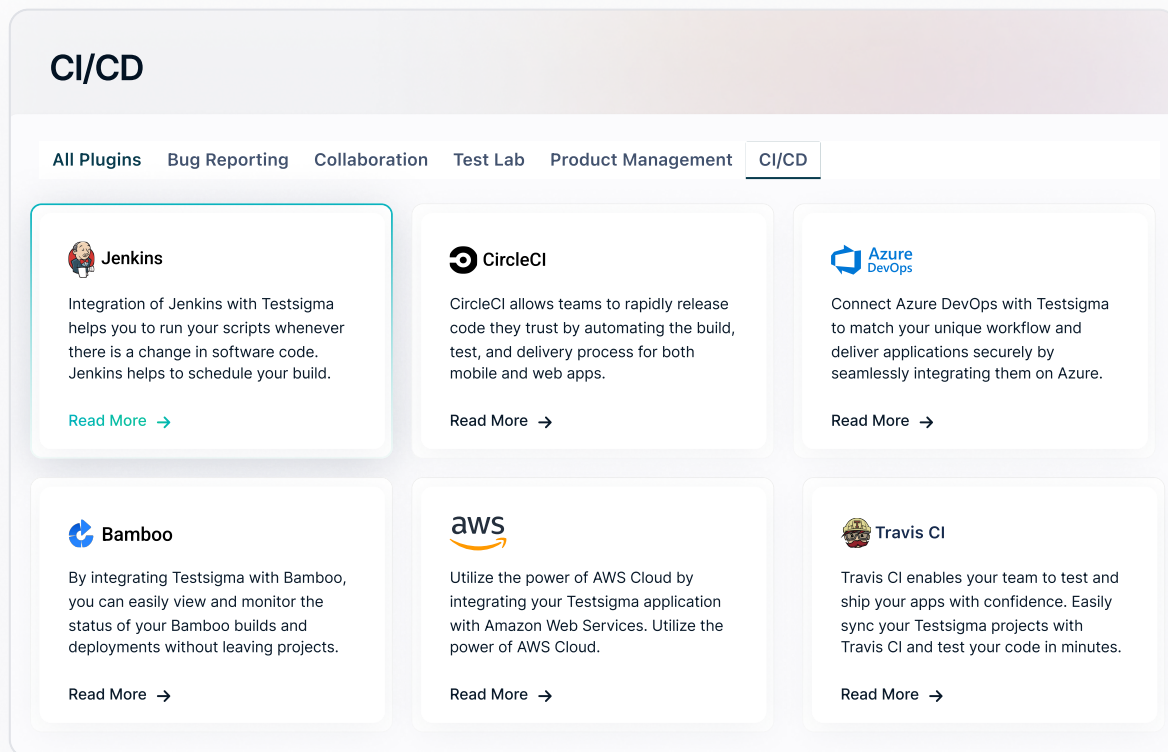
In the **U.S.**, Citibank users were locked out of mobile and credit card services during a widespread outage in 2025. In India, a UPI network failure halted nationwide digital payments for hours.

And in **Singapore**, a top regional bank faced repeated disruptions to ATM and mobile banking access.

In each case, automation pipelines passed. Yet critical issues still made it to production. **The average cost of a software failure in banking is \$2.3 million per incident**, not including reputational or regulatory impact. These incidents reflect a common theme: automation frameworks that validate code paths without covering business risk create a dangerous illusion of quality.

In just 26 months, nine major UK banks reported over **800 hours of IT outages** - many caused by software bugs that passed automated pipelines but failed in production.

Depending on the region, this callout can be replaced with examples like: the Citibank mobile banking failure in the U.S., the UPI network outage in India, or ATM disruptions at major Southeast Asian banks. These incidents continue to raise red flags with customers, regulators, and internal stakeholders alike.



The screenshot displays the 'CI/CD' section of the Testsigma interface. At the top, there's a navigation bar with tabs: 'All Plugins', 'Bug Reporting', 'Collaboration', 'Test Lab', 'Product Management', and 'CI/CD'. Below this, six plugin cards are shown, each with a logo, a brief description, and a 'Read More' link with a right-pointing arrow.

- Jenkins**: Integration of Jenkins with Testsigma helps you to run your scripts whenever there is a change in software code. Jenkins helps to schedule your build.
- CircleCI**: CircleCI allows teams to rapidly release code they trust by automating the build, test, and delivery process for both mobile and web apps.
- Azure DevOps**: Connect Azure DevOps with Testsigma to match your unique workflow and deliver applications securely by seamlessly integrating them on Azure.
- Bamboo**: By integrating Testsigma with Bamboo, you can easily view and monitor the status of your Bamboo builds and deployments without leaving projects.
- aws**: Utilize the power of AWS Cloud by integrating your Testsigma application with Amazon Web Services. Utilize the power of AWS Cloud.
- Travis CI**: Travis CI enables your team to test and ship your apps with confidence. Easily sync your Testsigma projects with Travis CI and test your code in minutes.

In each case, automation pipelines passed. Yet critical issues still made it to production. The average cost of a software failure in banking is \$2.3 million per incident, not including reputational or regulatory impact. **These incidents reflect a common theme:** automation frameworks that validate code paths without covering business risk create a dangerous illusion of quality.



3 Manual Testers: The Untapped Asset

Manual testers are closest to the real-world workflows that matter most. They understand the business logic behind a loan approval, the compliance nuances of KYC validation, and the edge cases buried in multi-step financial journeys. These are not just testers - they are domain experts, risk sensors, and customer advocates rolled into one.

And yet, they've been sidelined. Years of tool-centric thinking and code-first frameworks have made automation the territory of engineers. The very people who catch real issues during exploratory and UAT testing have been left out of the automation conversation.



It's time to flip that dynamic

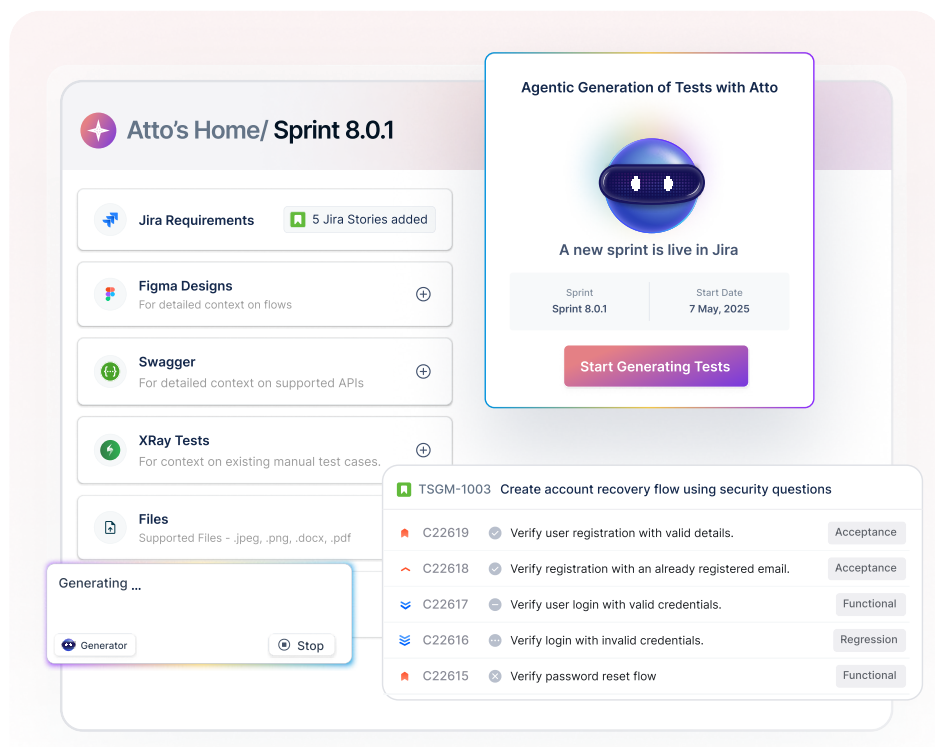
Manual testers have the intuition that no script can replicate. When given the right tools - tools that speak their language - they don't just participate in automation, they accelerate it. With natural language interfaces, AI-powered assistance, and agentic workflows, these testers can go from reactive defect reporters to proactive quality owners.

When testers are no longer forced to learn Java or debug XPath just to automate a login flow, they focus on what really matters: testing the things that break, not just the things that pass.

4 Enter AI + NLP:

A New Model for Regulated Banking

Agentic testing is a new approach that uses AI-powered agents to generate, execute, and optimize tests in plain English - made possible by advances in Natural Language Processing (NLP). This model doesn't require testers to learn code. It lets them describe the test they want in the language they already use every day - and an agent builds it.



This shift is especially powerful in banking, where workflows are complex, terminology is specific, and compliance scenarios are nuanced. With NLP, manual testers can express these scenarios naturally. The system interprets intent, generates structured tests, and adapts them as requirements evolve.

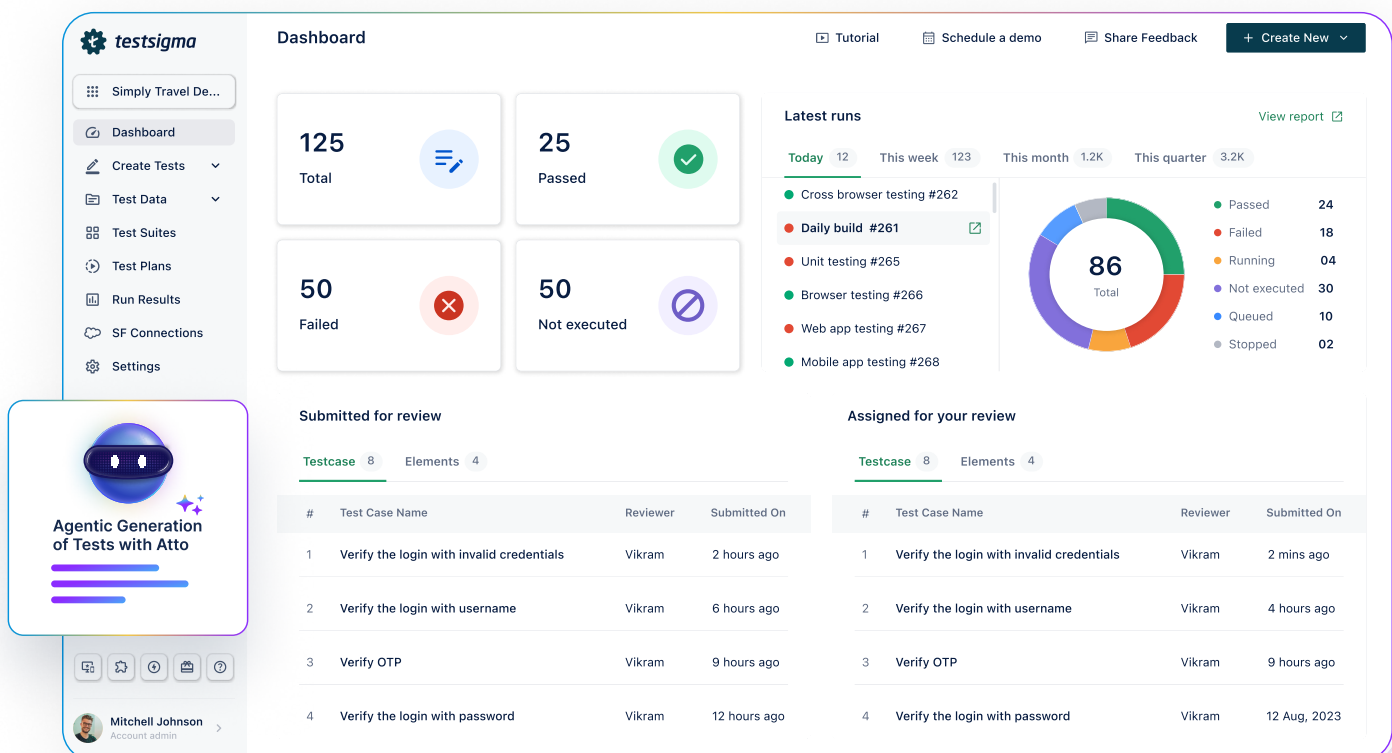
But the agents aren't autonomous in a vacuum. They are guided by people - by the manual testers and business experts who understand where risk hides. These testers act as mentors, pilots, and reviewers for the agents. They provide the business logic, domain insight, and real-world context that automation tools have traditionally lacked.

In this model

- ✓ Testers become test architects, guiding coverage with natural language.
- ✓ Agents act as execution partners, handling speed, accuracy, and maintenance.
- ✓ NLP bridges the gap between strategy and implementation - between compliance goals and test coverage.

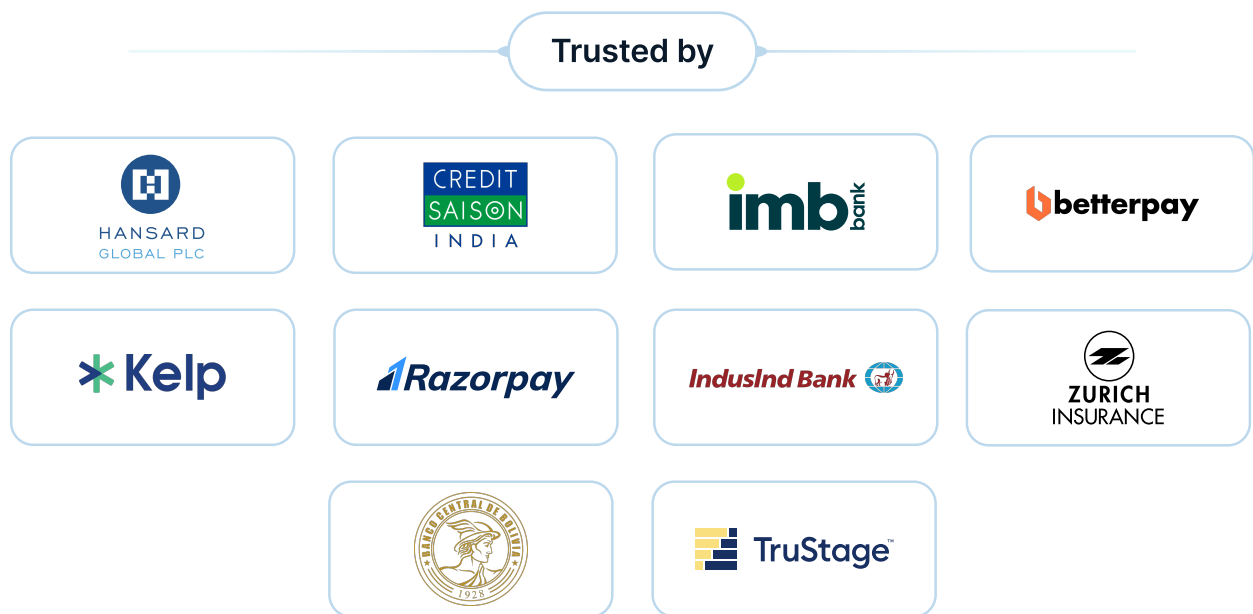
The result is dynamic, high-coverage QA that adapts as your systems, products, and regulations evolve. Instead of chasing green pipelines, your team validates real business outcomes.

This approach of combining AI and NLP doesn't replace testers - it amplifies them. It turns deep domain knowledge into scalable, testable automation. It enables QA teams to ship faster without sacrificing confidence. And most importantly, it restores control and visibility to the people who understand the business best.



5 Real-World Impact: Stories from Forward-Looking Banks

Several leading banks and fintech companies have already adopted this model. One team reduced regression testing from two days to two hours by enabling non-technical testers to write tests in English. Another expanded test coverage by over 50 percent without hiring additional QA staff. These organizations gained faster releases, fewer production bugs, and improved collaboration across QA, Dev, and compliance teams.



And 10,000+ QA teams worldwide

Testament of the Value Delivered

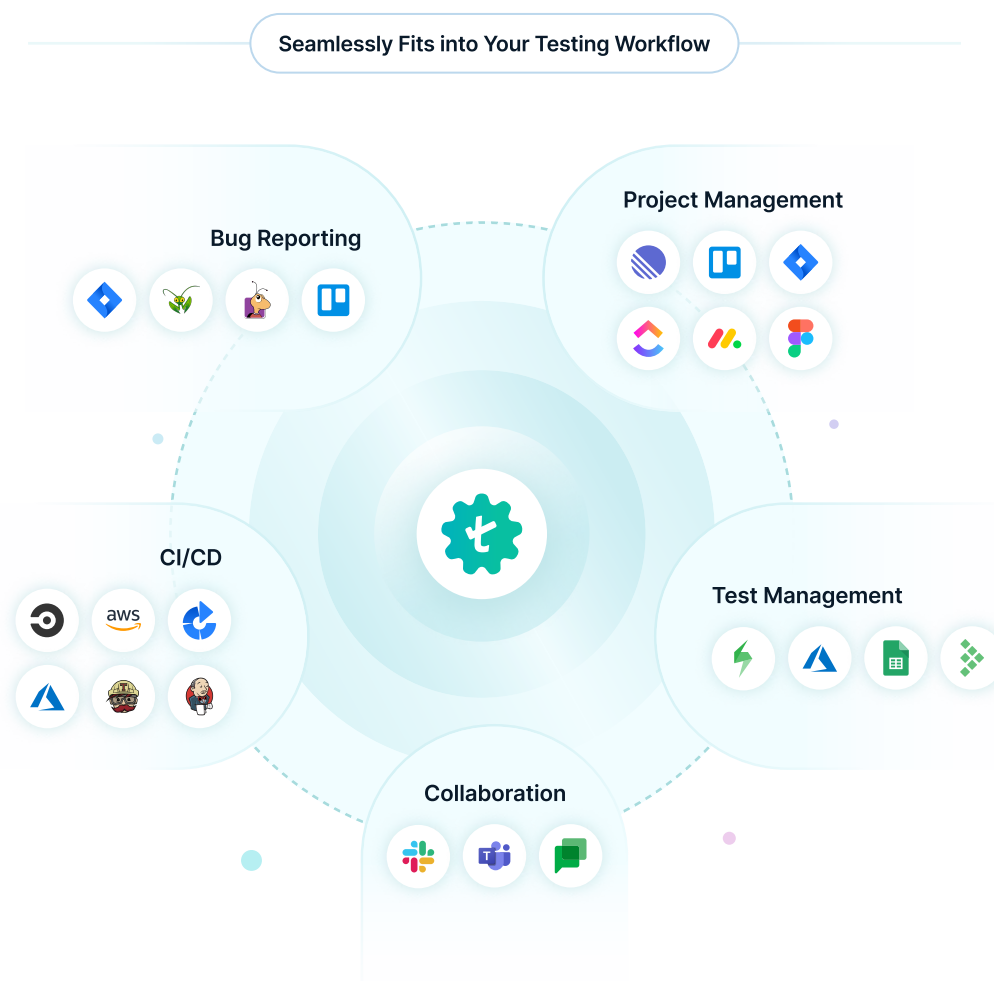
25M+
Tests Executed

10K+
QA Teams

70%
Reduced Testing Effort

6 Implementation Blueprint

You do not need to abandon your current tools to start adopting agentic testing. Begin by identifying workflows where manual testers can contribute. Introduce AI-powered test generation and natural language scripting alongside existing pipelines. Provide light training and early wins to build momentum. Within 30 to 90 days, you can expect expanded test coverage, reduced maintenance time, and improved visibility across teams.



7 The Future of Banking QA

Traditional frameworks are reaching their limits. They cannot scale with the speed of digital banking, nor can they adapt to modern compliance and customer demands. Agentic testing offers a path forward. It unlocks the potential of your entire QA team, not just the coders. It shifts QA from a reactive cost center to a proactive enabler of quality, trust, and velocity.

About Testsigma



Testsigma is an **AI-powered autonomous testing platform** built for fast-moving, regulated industries. With agentic testing capabilities, natural language scripting, and continuous optimization, it helps banks deliver high-quality software at scale without sacrificing compliance or control.



To learn more about **how Testsigma helps banking teams shift from frameworks to outcomes**, visit:



We have experienced quite a considerable reduction in our test execution time from 8 weeks to just 5 weeks per sprint

⚡ **75% Regression Suite Automated**



Holly Pennington
Test Manager



With Testsigma, we increased our test case creation speed by 400% for over 2500 automated tests. Its comprehensive device lab ensures we deliver high-quality releases every quarter.

⚡ **400% Test Automation Speed**



Sathish Babu
Sr. Engineering Manager



We chose Testsigma for its ease of use and flexibility. 80% of our test coverage is automated now, dramatically improving our software quality.

⚡ **80% Automated Test Coverage**



Jan-Clément Leloup
Technical Support Engineer

