TestResults

AUTOMATED SOFTWARE TESTING REPORT

2024/25

KEY INSIGHTS FROM PROFESSIONALS IN

banking, financial services, insurance, and MedTech

Table of Contents

This report provides a comprehensive overview of end-to-end testing and test automation practices, drawing on insights from professionals in regulated industries such as finance, insurance, and pharmaceuticals.

End-to-end (E2E) testing ensures software functions as intended—reliably and consistently.

Based on insights from test automation engineers, CTOs, and process specialists, this report examines how organizations address testing challenges, from managing legacy systems to coordinating diverse team structures. Key themes include fostering a quality-driven mindset, implementing effective test data management, and adapting to evolving automation tools and practices.

We invite you to explore these findings to gain insights into the current state of E2E testing and learn from industry leaders' experiences.

Methodology & data collection

To gather insights for this research, we conducted interviews with experts across different regulated industries, including finance, insurance MedTech, pharma, and banking. When talking directly to professionals in these fields, we were able to hear about their real-life challenges with test automation.

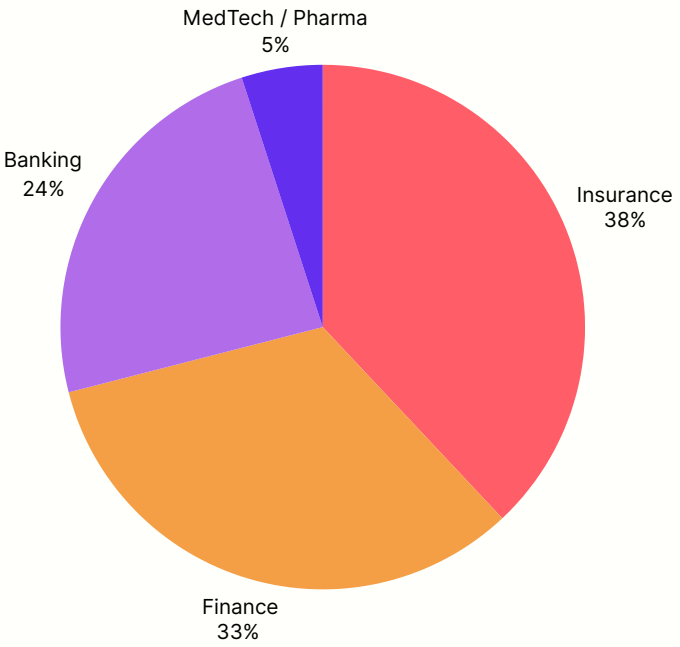
Expert interviews and perspectives

Our interviews provided insights from both IT professionals and process owners, each offering a unique take on E2E testing, particularly in terms of testing entire processes. These perspectives will be addressed in separate chapters based on the role of the individual within the organization—whether IT-focused or process-driven—offering a clearer understanding of the challenges faced.

Demographics

We spoke with individuals holding roles across IT and process management. This included senior positions such as Lead Quality Engineer, Senior IT Consultant, and Chief Technology Officer, as well as process-oriented roles like Head of Business Process Management and Director of Process Improvement. Their extensive experience offered valuable insights into how digitalization and testing are being managed within their industries.

Split of professionals by industry



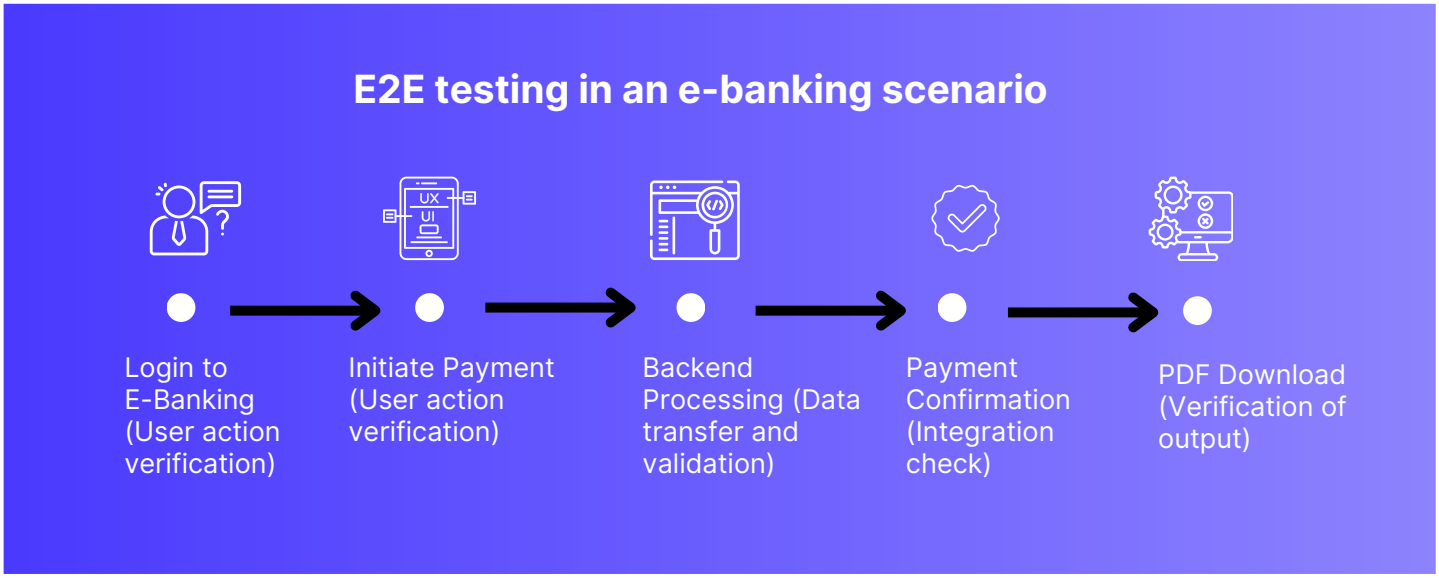
Understanding of E2E Testing Across Industries

The survey of professionals revealed diverse interpretations of E2E testing:

- Some define E2E testing as verifying a single application from start to finish, ensuring all features work correctly.
- The majority (70%) describe E2E testing as verifying that the entire system functions seamlessly, encompassing both frontend and backend interactions.
- Others emphasize testing across multiple applications or systems, focusing on inter-system communication.
- A subset highlights UI testing, emphasizing the importance of simulating real user behavior to validate the user experience.

At its core, **end-to-end testing verifies complete user journeys across multiple applications and devices.** It ensures that all system components—from user interface (UI) to backend systems, databases, and integrated services—work together seamlessly.

When referring to E2E testing in this report, we use the highlighted definition above.

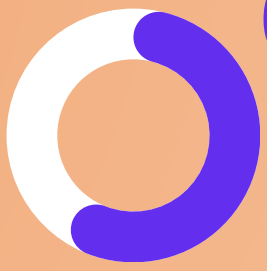


Many professionals believe E2E testing should replicate real-world scenarios, with around 60% emphasizing the importance of verifying customer journeys and business processes from start to finish.

E2E testing definition and importance

70%

of professionals described E2E testing as a way to test the entire system



60%

highlighted the importance of verifying customer journeys and business processes

TestResults

Key challenges in the automated E2E testing

The overall feeling about every company's own software quality is mixed, leaning toward positive but with some real frustrations. On a scale from 1-10, most rate their overall software quality between 6 and 9.

Older systems are holding strong after years of improvements, but the newer ones are struggling, barely hitting a 5. The issue? Teams are scrambling to get things out the door, and with testing resources spread thin, corners are being cut.

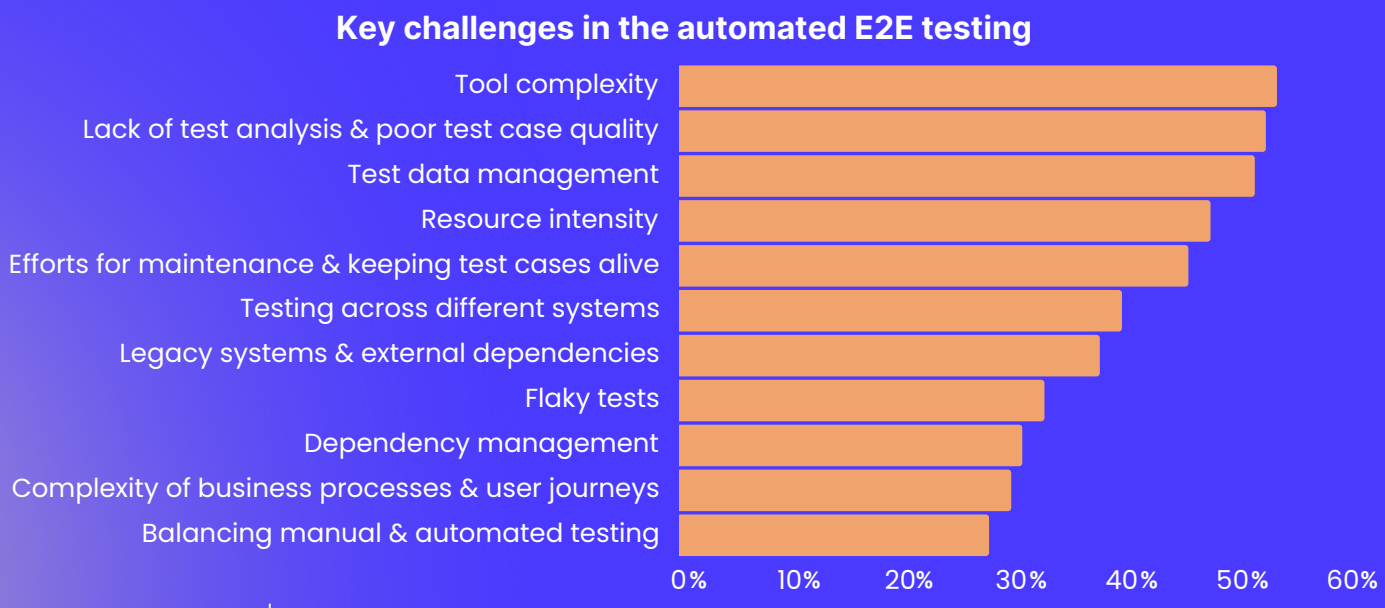
Average software satisfaction (1-10)

7.5

What's working well is full automation. Some teams have nailed it with E2E, unit, and integration testing fully automated. But keeping these tests running smoothly is a constant challenge. Any change in the software under test can cause headaches, requiring updates of the automated tests across the board. Everyone's expected to care about quality, but it's hard to balance speed and keeping things at a high standard.

Before diving into the challenges faced in end-to-end (E2E) software testing, it's important to understand the complexity and variety of issues teams encounter. Based on insights from industry professionals, several recurring themes emerged.

These challenges impact the efficiency and effectiveness of automated E2E testing efforts, highlighting areas where organizations often struggle:



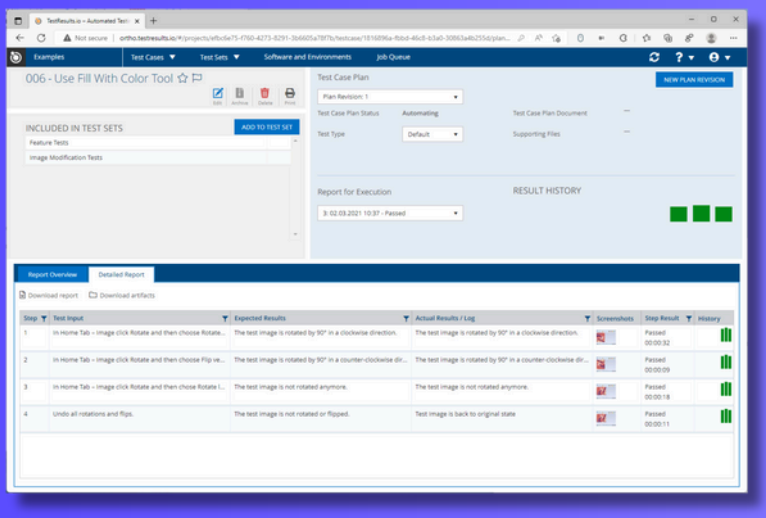
Tool complexity

53% of respondents find testing tools complex and hard to maintain. One professional shared that using Ranorex for automating Java-based interfaces led to constant issues, requiring frequent workarounds and support due to bugs. Achieving stable, automated testing across different applications proved challenging, as most tools aren't built for seamless E2E testing across multiple systems.

An expert noted issues with a Selenium-based suite: while developer-friendly, it wasn't user-friendly for testers, creating a collaboration gap. Since testers found the tool cumbersome, it led to delays and inefficiencies. Testers ended up spending considerable time troubleshooting the tool's issues instead of performing the actual testing, while developers found it hard to collaborate on these tests due to the tool's limitations.

Pro Tip: One of the biggest hurdles when choosing your test automation tool is the technical limitations. With TestResults.io, you are able to test user journeys across different applications and devices by emulating the user and therefore, being independent of any technology. Once automation is set up, it's reusable across testing needs and can be executed as a functional, a performance, or a stability test.

The platform is designed to simplify test automation, execution, and maintenance, so you spend less time on all three and more time focusing on delivering quality software. TestResults.io helps you handle it all with ease, reducing time to market and ensuring that automated tests are stable and dependable. It also cuts down false positives and flaky tests by 99%. To learn more, visit the website testresults.io

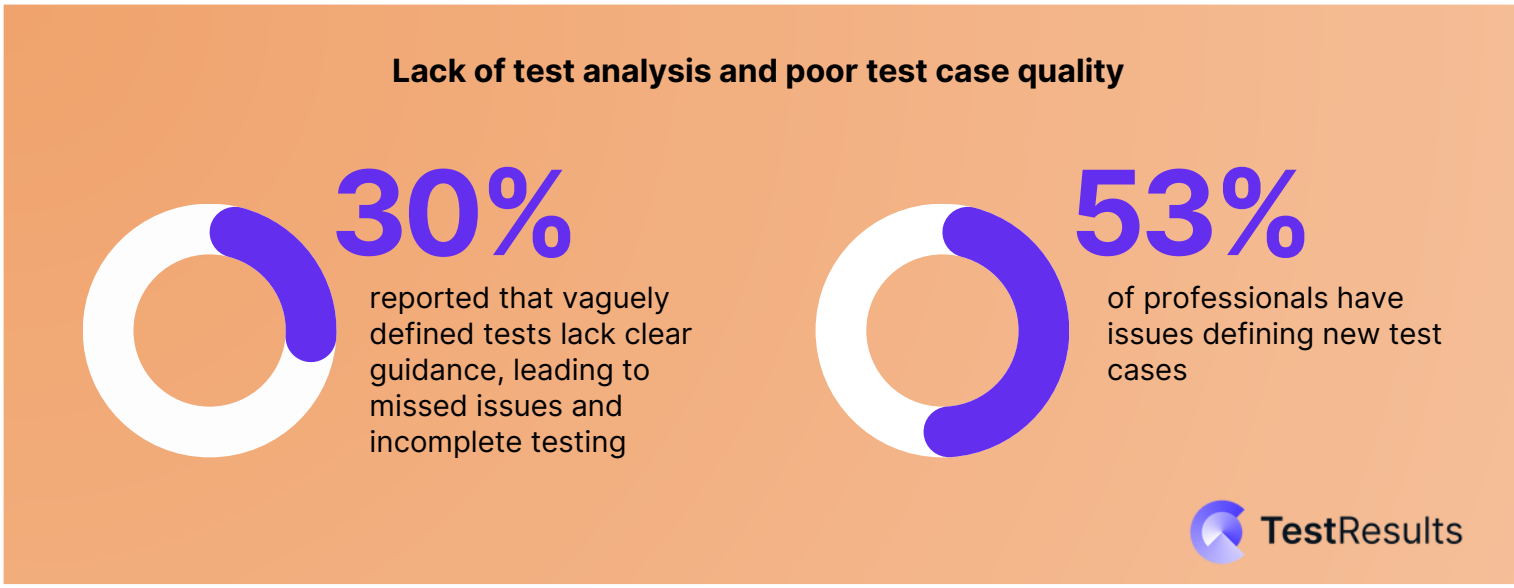


TestResults

Lack of test analysis and poor test case quality

53% of professionals highlighted the challenge of defining proper test cases for automation. Many mentioned they face issues when defining new tests, leading to a lot of wasted efforts—cases are outlined but then discarded.

Another **30%** pointed out that poorly defined regression tests often lack the clarity needed for effective use causing confusion and inefficiencies.



Test data management

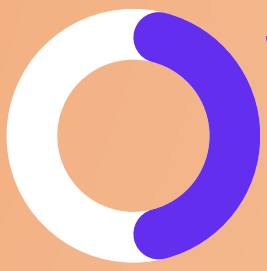
52% reported that **test data management** is often a significant bottleneck. The challenge lies in setting up the specific data configurations needed for complex applications without using sensitive customer data.

30% pointed out that older legacy systems—such as outdated databases or company software—store data in formats that are hard to replicate. This makes it difficult to create stable, reusable test data for testing environments. Since these environments require realistic, consistent data to simulate real-world conditions, the inability to recreate such data from legacy systems significantly hampers testing accuracy and efficiency.

Challenges in test data management

30%

noted that legacy systems store data in hard-to-replicate formats, limiting realistic test data and reducing testing accuracy



50%

of professionals face issues with test data management

TestResults

Resource intensity

About **48%** said that high-quality software testing **is very resource-intensive**—both in terms of time and people.

Verifying complex business processes often requires significant manual effort. Teams end up spending a lot of time running repetitive tests, which can pull testers away from more valuable activities like exploratory testing—essentially keeping them from tackling the fun stuff.

Software testing demands significant resources

48%

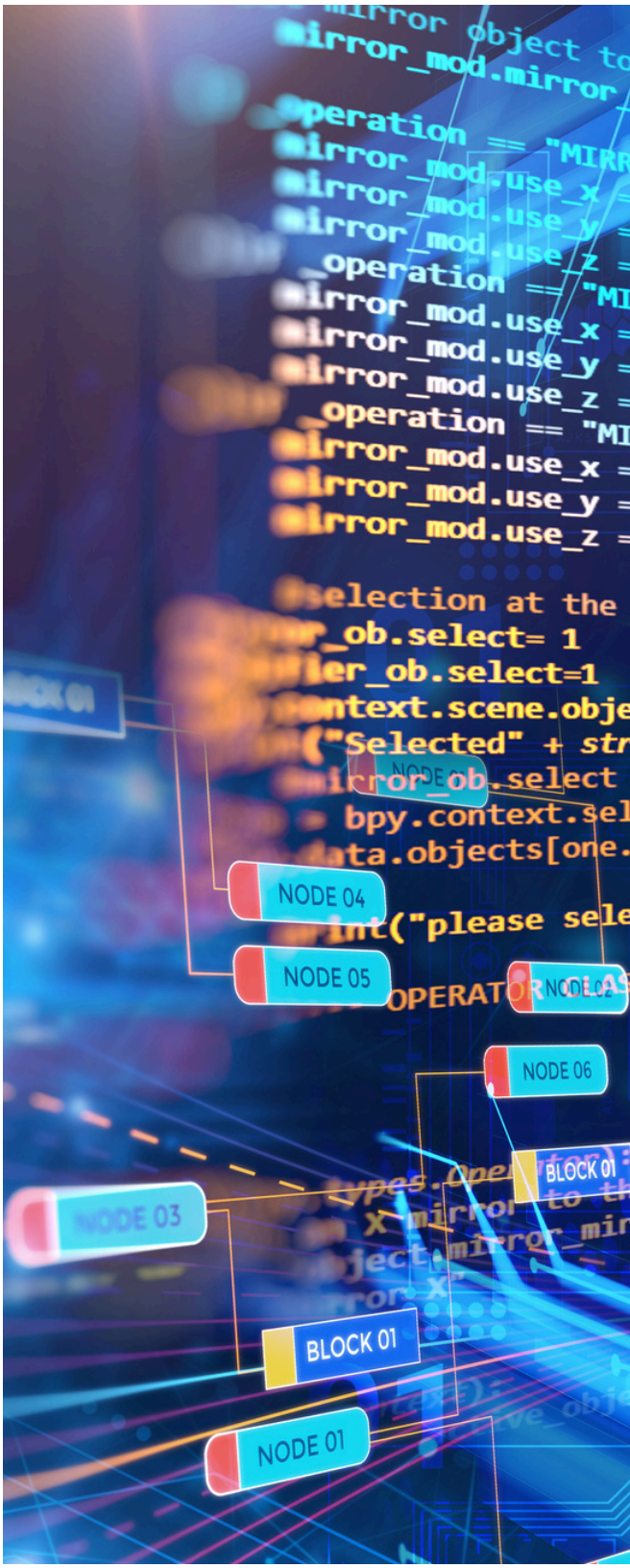
professionals said that high-quality software testing is very resource-intensive

TestResults

Maintaining tests also poses a real challenge. IT departments must ensure that all changes after updates are reflected in the automated test cases, and adjust identifiers if needed to avoid flaky tests.

This means the one responsible for the automated tests must sift through and update test cases, which is time-consuming and often leads to gaps in test coverage. Due to the time required, teams may miss critical updates, leaving some issues unaddressed and compromising software quality.

Many companies still struggle to automate E2E testing across different applications, leaving teams stuck with manual testing, which consumes significant time and resources. As one expert noted, being forced to do E2E testing manually highlights the need for better automation tools to speed up the process and achieve full coverage. However many available options are technologically limited, preventing teams from fully automating entire user journeys. and therefore the real-life scenario.

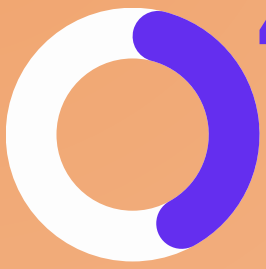


Efforts for maintenance and keeping test cases alive

Maintenance of test cases is an ongoing effort, with 3 or 4 full-time employees focused on exclusively this, depending on how dynamic the application or processes are. **46% of experts mentioned that frequent changes in element IDs or UI layouts significantly increase their maintenance workload.**

Another **30% talked about the problem of flaky tests**—false positive tests—which fail even though there is no bug in the application. Debugging these flaky tests is a time-consuming process and often leads to unnecessary efforts.

The Test Automation Maintenance Trap



46%

say frequent changes in element IDs or UI layouts greatly increase maintenance workload

30%

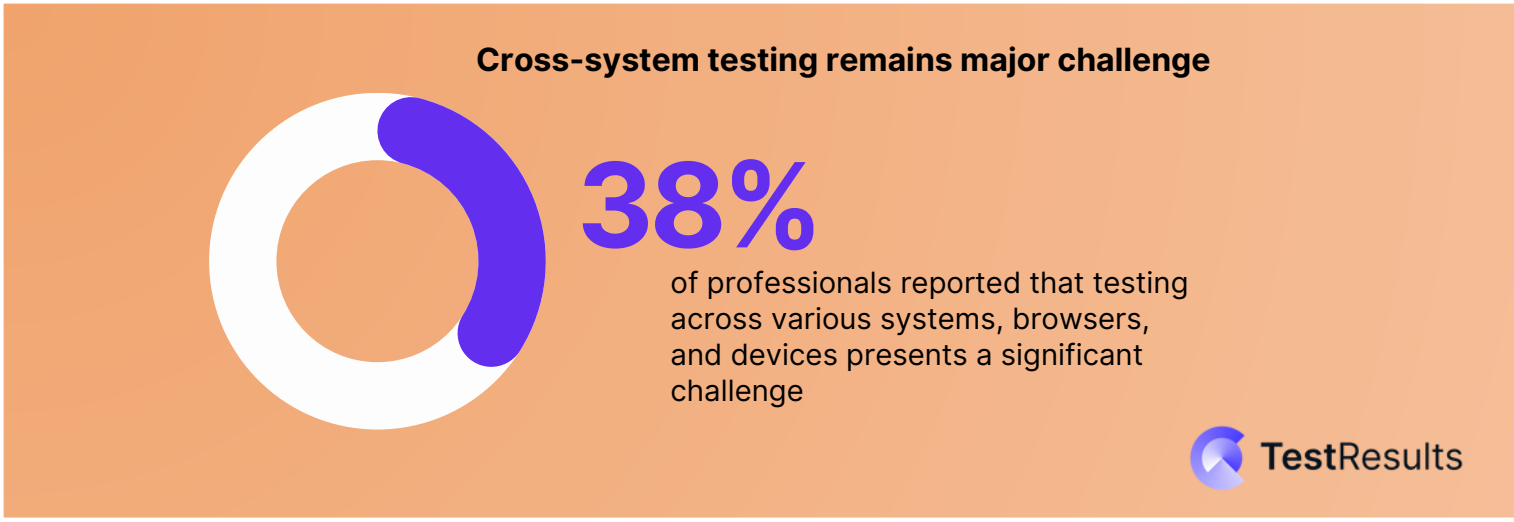
talked about the problem of flaky tests

TestResults

Testing across different systems

Around 38% of professionals mentioned that **testing across different systems, browsers, or devices is a big challenge.**

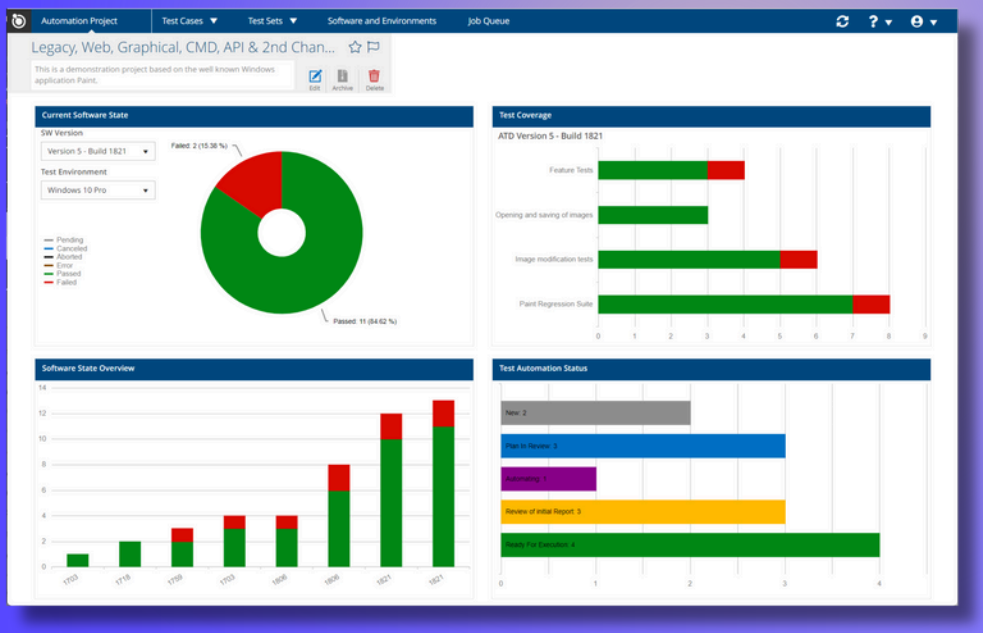
One explained that automating testing for complex systems is particularly difficult. Currently, the entire workflow requires manual testing. Although there's a push toward automation, the way these systems are structured makes automation challenging. Different systems often rely on unique configurations or protocols, meaning automated tests struggle to cover every aspect reliably without frequent breakdowns. Thus, testing manual is the more viable but time-consuming option for now.

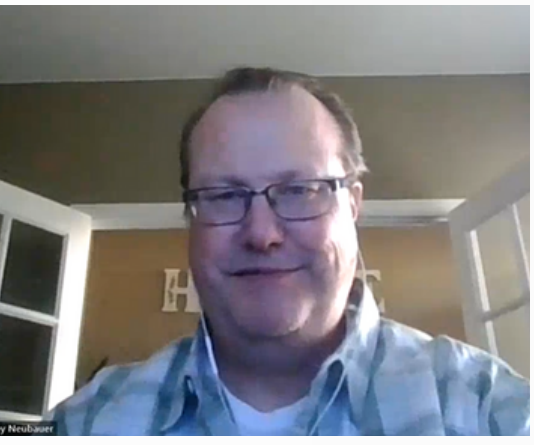


An expert pointed out just how brutal it is to coordinate tests across systems built by different vendors and using mismatched technologies. You're left testing isolated parts of the workflow, with full E2E verification often getting overlooked—opening the door for critical issues to sneak through.

Pro Tip: Use TestResults.io's modular test design to manage complex user flows without getting bogged down by each system's unique setup. You can adapt tests to each system's quirks without starting from scratch every time.

This approach lets you zero in on key user flows across multiple browsers, applications, and devices, cutting down on maintenance time and boosting test coverage. To learn more, visit the website testresults.io





"I'm not spending all my time refactoring code, updating test cases and everything. I'm spending more time actually out there testing what's supposed to be tested as well as maintaining a great test case baseline for regression and other testing if needed.

Safely I can say it's 90% faster with TestResults.io than Selenium."

Bradley Neubauer, Lead QA Engineer at Publicis Sapient
Police & Government USA

Unclear requirements and coordination issues

30% noted that managing dependencies between developers and testers, along with ensuring high-quality requirements, pose major challenges.

They emphasized that coordinating testing efforts between multiple teams is a significant hurdle, especially when several systems or products are integrated. This makes it more difficult to achieve proper alignment.

Notably, while respondents could report the size of their development teams, few knew how many were involved in testing. Usually, **there is a 1/10 ratio between testers and developers.**

This points to a disconnect between development and testing efforts, underscoring the need for having enough testers in the team and the visibility across these teams.

Legacy systems & external dependencies

38% mentioned that certain systems and applications, especially those using older proprietary technologies or external microservices, **are difficult to test.** This difficulty often arises because teams lack access to the underlying code, making it challenging to create effective test scenarios. Without access to the code, testers can't modify or inspect the system's inner workings, which restricts their ability to automate tests or pinpoint issues precisely.

Additionally, external microservices often rely on third-party providers, meaning changes or instability in these services are beyond the team's control, further complicating the testing process.

Additionally, **32% noted that older modules or external components are beyond their control,** allowing only manual testing or high-level validation. These systems, often managed by third parties, limit teams' ability to automate tests or make adjustments, making manual testing time-consuming and resource-intensive.

Legacy systems and complex applications hinder test automation

38%

mentioned that certain systems and applications are difficult to test

32%

noted that older modules or external components are beyond automated control

TestResults

Complexity of business processes and user journeys

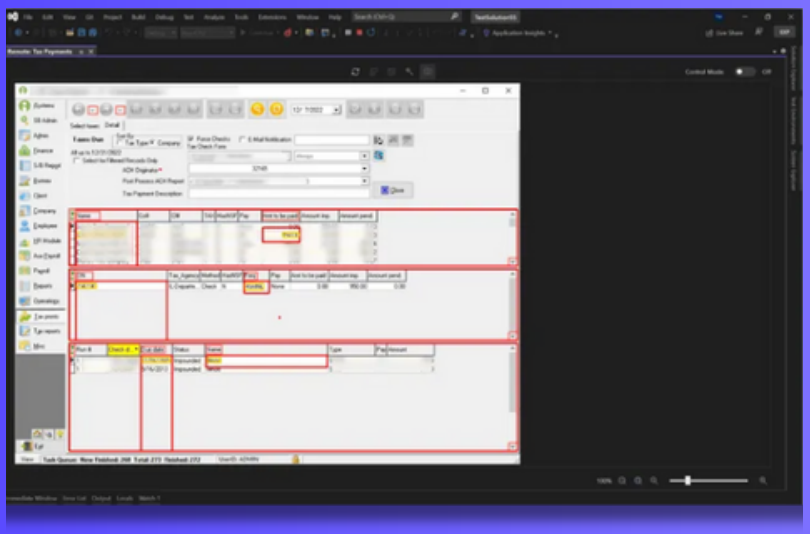
31% stated that the complexity of business processes makes E2E testing particularly challenging, as there are multiple technologies and applications involved.

One expert mentioned that a typical business process involves more than 25 different applications. Coordinating tests across multiple applications in a complex business process can be a logistical nightmare.

E2E testing is essential to make sure everything works together, which can be extremely time-consuming and prone to flaws when done manually. Automated testing not only saves time—it's the only way to stay sane when managing such complex, interconnected processes. On the other hand, **the wide variety of different technologies involved in these processes makes it challenging to automate the testing of these digital business processes or user flows.**

Pro Tip: With a customer-centric test automation tool like TestResults.io, you can automate testing throughout the entire digital business process. TestResults.io emulates the user, so you can automate across different applications and technologies with the same approach.

If you have ever wished for stable, reliable automation, reducing false positives and minimizing the need for hands-on checks, this might be for you. Saves you up to 90% of automation effort compared to Selenium, according to our customers. To learn more, visit the website testresults.io



Balancing manual and automated testing

About 30% of professionals cited difficulties in balancing manual and automated testing. Some user flows involve complex interactions, such as those with delays in data exchange between systems or uncertain timing of data updates across platforms. For example, in cases where one system takes several minutes to receive or confirm data from another, it becomes difficult to automate tests accurately, as the exact point of data synchronization can be unpredictable

One respondent explained that manual testing is essential for workflows involving third-party integrations or physical components. These areas often can't be fully tested with automation, as manual checks ensure accurate validation where automation may fall short. Interviewed companies generally aim to use automation for efficiency and speed but rely on manual testing for reliability in complex or unpredictable scenarios.

Process owners and E2E software quality

According to the definition used in this report—the entire user flows across different applications—E2E (end-to-end) testing is, at its core, testing a process. While testing, particularly automated testing, is typically the responsibility of the IT and/or QA department, it is worth examining E2E testing and process quality from the perspective of the process owner.

Therefore, we interviewed process owners from the same industries. 60% reported that the process management function is embedded within the IT department. A further 15% distribute process managers across various departments, while another 15% report directly to the office of the Chief Executive Officer (CEO). Irrespective of the organizational structure, process owners and managers consistently rely on the IT team as the requesting party.

Significant investments in digitalization, **encompassing the integration of increasingly more software into core business operations and processes, are ongoing,** with annual expenditures ranging from $1 million to $30 million.

However, process owners lack visibility into the holistic health of their processes. Disparate dashboards and reports are disseminated across development, testing, and business teams, necessitating process owners to rely on feedback from multiple stakeholders. While some companies reported that this approach works reasonably well without major issues, others indicated that the volume of bugs is unacceptably high.

The processes themselves are inherently complex, leveraging multiple systems and interfaces. Even minor changes can have unpredictable ramifications across other process components. This very complexity presents a significant challenge in automating testing of digital processes.

As the cadence of feature releases continues to intensify and more workflows become digitalized, **the question arises whether there is a need for a centralized, consolidated view of the company's software health and the digital processes across it.** Such a holistic perspective would provide access for process owners, IT, and the broader business, enabling informed oversight and proactive management of the organization's digital ecosystem.

The diverse landscape of test automation

46% said that **choosing the right automation tools** is crucial, and ensuring they fit well with their existing processes is even more critical. These are three different types of test automation tools:

1. ID-based solutions, which interact with the software under test through identifiers on the code layer
2. Visual or graphic solutions, which rely solely on the user interface.
3. Customer-centric approach, which emulates the user and interacts with the software like an end-user would.

Ensuring the selected automation tools seamlessly integrate with the organization's existing processes and landscape is cited as a top priority by the majority of respondents.

ID-based	Visual	Customer-centric
+ **Quick execution:** Tests run fast once set up, saving time.	+ **Functionality focus:** These tools handle functional testing, ensuring that visual elements respond correctly to user interactions, like clicking buttons or submitting forms.	+ **Low maintenance effort:** Once tests are automated, you don't have to put too much effort into maintaining them.
+ **Community support:** Extensive online resources and forums for troubleshooting and guidance.	+ **System-agnostic:** Compatible with various systems, offering flexibility across platforms.	+ **Technology independence:** Checks if the entire user flow is working as expected, finding bugs that might have slipped other way.
+ **Efficient cross-browser testing:** Simplifies testing across different browsers.		
– **Time-consuming setup**: Mapping all elements, especially in complex apps, can take a while.	– **Prone to false positives:** Minor visual changes can trigger unnecessary alerts, requiring lots of maintenence effort.	– **Takes time to set up:** Building test cases that reflect real user journeys requires careful planning.
– **Requires coding skills**: You need technical knowledge to use these tools effectively.	– **Slower execution:** Visual checks can take longer to run compared to other testing methods.	– **Potential lack of clarity in technical issues:** Some technical issues may be harder to pinpoint with user-centric testing approaches.
– **Sensitive to changes**: Frequent app ID changes can lead to test failures and constant updates.		

5 critical factors undermining effective test automation

Testing is a crucial part of high-quality software development. When starting early, using thoughtful automation, and encouraging team collaboration, organizations can continuously improve and deliver better results.

Here are some key lessons and best practices for software testing, based on insights from industry experts:

1. The organizational challenges of testing ownership and responsibility

Research indicates that testing responsibilities within organizations often lack clear delineation. While quality assurance traditionally falls under the purview of development teams, high pressure to deliver new features frequently results in testing being delegated to business units or dedicated manual testing teams.

Development teams often prioritize feature development over implementing and maintaining automated tests due to tight delivery schedules and resource constraints. This gap in testing coverage is typically addressed by business unit personnel or dedicated manual testing teams. However, this shift in responsibility creates additional challenges, as business users usually lack the technical expertise required for test automation implementation and maintenance.

One finding indicates that there is often limited visibility across testing activities within organizations. IT teams frequently lack comprehensive understanding of who is involved in testing processes across different departments, leading to potential redundancies and gaps in test coverage. This fragmentation of testing responsibilities can result in inconsistent quality assurance practices and inefficient resource allocation.

Furthermore, this decentralized approach to testing often impedes the implementation of standardized testing practices.

2. Building a quality-first culture

While 39% of professionals believe that software quality is everyone's responsibility, the C-suite often views high-quality output as a given. They prioritize rapid development, assuming that quality assurance will simply fit into place without additional resources. This focus on scaling development often overlooks the need to scale the QA team accordingly.

Rather than understanding testing as an integrated responsibility, many C-level leaders push for developers to handle unit tests, assuming this will ensure accountability. Testers, then, are expected to focus on larger, more complex testing tasks like integration and E2E testing. This compartmentalized approach, however, can miss the cohesive quality management needed across the development lifecycle.

38% talked about how important it is for developers, testers, and business stakeholders to work together. When everyone sees quality as a shared goal, it's easier to achieve high standards and reduce the number of issues that make it to production.

3. Regression testing: Barriers to automation in modern software development

As software complexity increases with each release, the scope of required retesting expands.

Regression testing—the process of verifying existing functionality following system modifications—is essential for maintaining software integrity. However, without robust test automation, regression testing can rapidly become an unmanageable burden on resources.

When regression testing requirements exceed capacity, teams become overwhelmed by the volume of manual verification tasks. This not only impedes release cycles but also consumes valuable team resources that could be allocated elsewhere. Automation of regression tests emerges as a critical solution, particularly given the impossibility of predicting the impact of new features or modifications.

Most teams lack sufficient capacity to conduct comprehensive manual regression testing. Consequently, they often implement partial testing strategies that may not provide adequate coverage. Our research indicates that when regression testing remains manual, three primary factors are typically responsible:

- **Development teams prioritize new feature development** over implementing test automation due to resource constraints

- **Knowledge gap in test automation capabilities:** While business units often conduct manual regression testing due to developer resource constraints, they typically lack the technical expertise to implement and maintain automated testing frameworks

- **The immense time investment required for implementing and maintaining automated tests** leads organizations to continue with manual testing approaches

4. Tool fragmentation in automated testing landscapes

Most automated testing tools are designed for specific application types or technologies. Consequently, organizations must implement multiple testing solutions to achieve comprehensive coverage across their application landscape. This diversification necessitates significant investment in various aspects:

- Multiple license fees (for proprietary solutions)

- Specialized automation experts proficient in each tool

- Dedicated resources for tool administration (including implementation, maintenance, updates, and integration)

Furthermore, technological limitations present significant challenges when testing peripheral systems and third-party applications using conventional ID-based automation approaches. These constraints restrict the scope of automated testing capabilities and add complexity to the overall testing strategy, potentially compromising the effectiveness of quality assurance processes. As a result, end-to-end testing across multiple applications remains largely manual, as the popular ID-based automated solutions struggle to bridge the technological gaps between different systems effectively.

5. The hidden costs of test automation maintenance

Test automation maintenance, particularly addressing unstable test cases, represents a significant resource investment. If maintenance efforts equal or exceed the time required for manual testing, the automation initiative loses its operational value.

Organizations typically dedicate 1 to 3 Full-Time Equivalents (FTEs) to maintaining automated test suites, which often undermines the expected efficiency gains. Therefore, test stability and maintenance requirements should be primary considerations when evaluating automation tools or assessing the effectiveness of existing automation frameworks.

Persistently unstable tests ultimately lead to diminished confidence in test results and reduced motivation to maintain the test suite. This deterioration often results in the abandonment of automated testing efforts within months, forcing organizations to revert to manual testing processes.

Conclusion: Advancing software quality amidst digital transformation

As digitalization reaches new heights, the pressure to deliver more features at a faster pace has grown tremendously. Yet, this rapid evolution has brought increased complexity in IT systems, making software quality harder to maintain. Testing now faces heightened challenges, with automation not advancing as quickly as it's needed to keep pace. Key issues—such as technological dependency and lack of stability—continue to hinder progress.

TestResults.io was founded with the mission of elevating software quality on a global scale by empowering companies to implement stable, cross-technology, cross-device test automation. Our approach boosts efficiency by cutting up to 74% of automation time and minimizing maintenance efforts with built-in stability. We are poised to make high-quality software achievable even in the face of increasing digital demands and scares ressources.

Learn more on how to get yourself this unfair advantage in software quality.