AVO™

Automation

Test Automation Strategy For Beginners

Index

Introduction

The software development world is impatient. On the one hand, the development cycles are getting shorter, while software quality expectations are getting higher. Add agile and DevOps methodologies to the mix, and it becomes clear why software developers are taking to automated testing.

The conventional waterfall development model worked well for an era when development companies had the luxury of time. But in today's uber-competitive market, with compressed CI/CD pipelines, manually testing different aspects of the software in the last phase of development doesn't make the cut.

Manual testing isn't just slow. It's tedious, requires a lot of human resources, and is error-prone. Testers are forced to work on repetitive tests over hundreds of hours instead of focusing on the business-critical aspects of the software. The Systems Sciences Institute at IBM reported that the cost to fix bugs found during the testing phase could be 15x more than the cost of fixing those found during design. Not only this, but manual testing is also responsible for letting several bugs slip through in the released software product due to time constraints and human errors. This can lead to massive technical debt, unwarranted downtime, reputational damage, and unprecedented financial losses.

Just look at the example of the Knight Capital Group. A technical glitch in its trading software cost the firm $460 million! It lost 75% of its market value within a day and later went on to be acquired by competitor Getco LLC.

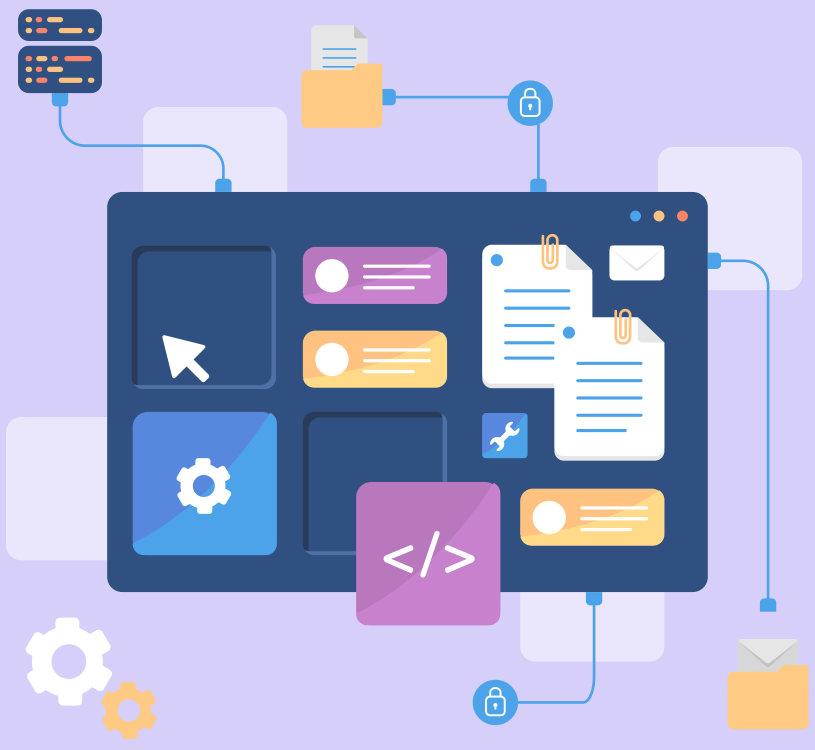
Incidentally, automated testing can resolve most, if not all, of these issues. It gives better test coverage, brings down testing costs, shortens the testing cycle, identifies and corrects 90% of all bugs, and fits right in with the mantra of continuous integration, continuous development, and continuous deployment.

Almost a quarter of respondents in a survey claimed that they saw returns on their test automation investments almost immediately or within six months.

Curious?

So, let's dive right into it!

What is test automation?



Simply put, test automation comes under the purview of the quality assurance and testing departments. It enables running tests on the software being developed using computer programming and coding without any manual interference. It's automatic, fast, and can provide highly reliable results. It's usually used in conjunction with manual testing and not as a complete replacement. Tests that are repetitive, hard to configure, and determinant are automated to make the entire testing process much more manageable. Automated testing also plays a crucial role in the continuous testing requirements of agile and DevOps methodologies.

With the rise in the popularity of test automation, many testing tools and solutions have also made their way to the market. These test solutions help control the execution of the tests using test cases and scenarios. The displayed results are then compared to the expected results to identify errors, bugs, and flaws.

What are the benefits of test automation?

Test automation, if implemented effectively, gives massive returns while developing products. From decreased costs to faster delivery, there are five key benefits to automated testing.

Lower costs

Every developer worth their salt knows that the most cost-effective way to fix a bug is to do it as early in the development cycle as possible. This requires two things. First, the bug needs to be identified early on. Second, it needs to be immediately fixed. This is where test automation comes in. By continuously running automated tests, developers can find and fix bugs as soon as they compile their code. Instead of waiting for the last phase of the SDLC, the bug gets fixed early, saving the firm almost 75% of the cost they would've otherwise spent fixing the bug.

Test automation also reduces the number of testing hours required. While manual testers can only work 8 hours a day, computers can run the tests 24/7. This effectively reduces the number of hours of paid testing work you require from your employees.

Better test coverage

Manually increasing test coverage is tricky. With new features being added to software regularly, manually testing them becomes a tedious task. With no-code test automation solutions, you can test the growing applications with a few clicks of buttons, enhancing the test coverage. With regression testing taking the longest time in testing, doing it manually is no longer feasible. Automation makes it a breeze, in turn improving the test coverage.

Better test accuracy

One of the most significant issues with manual testing is the mind-numbingly repetitive task of going through the same tests hundreds of times. This eats into the productive hours of your testers and promotes a pattern of boredom and distraction. And with distraction comes mistakes, ergo a bug-ridden software that glitches constantly. Automated tests can run repetitive cycles on the trot without stopping and still give the same accurate results.

Faster delivery

In a competitive market, time-to-market is everything. Even a day's delay can mean a loss of significant market share. In such a scenario, shorter testing cycles are necessary. If you spend five days in debugging and regression testing in a standard two-week sprint, the same can be reduced to two days with test automation.

Easy reporting

A robust test automation software will have inbuilt intelligent reporting capabilities giving you access to metrics and data that aren't usually visible. These visually stimulating reports can help uncover patterns and bugs quickly. Moreover, all the test results can be viewed on a single dashboard.

How to get the most out of test automation?

The jump from manual to automation testing can be beneficial, but only if done right. A Capgemini report showed that 41% of IT teams do not possess the proper methods, frameworks, and strategies to implement automated testing effectively. While automation can save time and money, it can also cause chaos, decrease potential returns, and add to the developer's woes. The difference between the two scenes lies in building a suitable test automation strategy.

Here's how you, too, can design the most effective testing strategy for your product.

Define your testing goals

One of the most naive mistakes most QA teams, new to automation, makes is automating everything. Not all tests are meant to be automated. It's essential to identify the tests that provide the most ROI for the least effort and work on automating those only.

This is the step where you define the scope of testing and select the tests you wish to automate.

Automation works best for tests that are:

1) Repetitive
2) Risky
3) Require large data sets
4) Critical to the business
5) Require testing across different platforms, browsers, and configurations
6) Determinant in nature
7) Not viable to execute manually
8) Involve no creative logic

And then some tests should never be automated, like:

1) GUI tests
2) UX tests
3) Tests that are to be executed only once
4) Tests that are non-determinant
5) Tests that require visual perception
6) Tests that work with unstable functions
7) Ad hoc and exploratory tests

This brings us to the next point.

Find the ideal testing approach

Based on the pointers given above, here are the tests that should be automated because they guarantee high returns.

Unit tests

These tests form the core of the strategy. Ideally, they're written at the beginning to check the individual pieces of code for bugs.

Regression tests

Manual regression testing takes a long time. This is one of the tests that should be automated at all costs.

API tests

Testing APIs helps validate the software's behavior and logic. Since most applications today rely on API, it makes sense to automate these tests, especially in cases where there's repetition.

Cross-browser tests

With so many devices, platforms, browsers, OSs, and screen resolutions on the market, it's critical to check whether your application works perfectly on each possible configuration

Data-driven tests

Manually entering data into data fields in your application in every possible combination can be a headache for testers. Test automation can finish this task quickly and accurately, generating the required results.

Smoke tests

Smoke tests are used to generate instant feedback about critical functionalities such as configurations, permissions, and dot net frameworks.

Select the right test automation framework

A test automation framework can be defined as detailed guidelines that dictate how the test will be written and run. They help streamline the entire testing procedure. There are a few frameworks to choose from.

Linear

In the linear approach testers, the most basic framework of all performs each step manually and records them with an automation tool.
While it's fast and straightforward, it's also erroneous and hard to maintain.

Library architecture

In this framework, the common tasks are grouped and stored together in the library, making them easily accessible.

This framework is cost-effective and scalable but also requires constant updates to the script whenever the data changes.

Modular

The modular approach divides the application into different modules and tests them in isolation.

While it makes it easy to test for modular changes, it does not work well with multiple datasets.

Data-driven

Unlike linear or modular frameworks, this approach doesn't hard code the data but stores it externally. This way, the tester can pull in different datasets without having to rewrite the script.

Keyword-driven

This approach is primarily used to test the GUI of the applications. The keywords help create a string of actions that can be seamlessly tested automatically.

This approach works well since the code is reusable. But it also requires an expert coder to create the keyword repositories.

Hybrid

This approach combines all the other frameworks to magnify the advantages of each and minimize the flaws. It is the most recommended approach for any development company.

Choosing the right automation solution

This is a critical step. The wrong test automation software can derail your project and set you back on your deadlines. When picking the testing software, there are a few points that you need to keep in mind.

- Ensure that the test automation software works well with the tech stack being used in the project.
- Also, look for the mobile platforms supported by the software. Preferably pick a solution that supports both iOS and Android.
- The platform should be easy to use and have features such as no-code and keyword libraries. But just because it's easy to use, there should be no compromise on the quality and high-level functionalities of the platform.
- Most companies are working on shoestring budgets. Splurging unnecessarily on an automation tool isn't advisable for them. That's why also check your software for affordability and expected ROI.
- Also, consider knowledge resources provided by your test automation partner and their customer support.

Ensuring the team is 'no-code' ready

Now that you've chosen the tests you want to automate, picked up a framework, and have access to the right automation solution, it's time to build the tests and run them. Using a no-code automation solution, you can easily design and deploy tests without having to spend hours scripting.

What many companies do to cut corners is force their manual testers to build test cases. This is not an ideal way to deal with test automation. Test automation is a vast field in itself and requires an expert to ensure the best possible returns.

Other pointers to keep in mind here are

- Prioritize critical test cases first. This way, you'll ensure no time is wasted on the wrong tests.
- Build the test case concisely and simply to make it understandable for other testers.
- Also, build the test case in a way that makes it easy to reuse time and again.

Reusing the test cases

Test automation has the potential to give good ROI because of its reusability. This feature comes in handy, especially during extensive regression tests.

One way to ensure reusability is to create small test cases. Instead of building one case for a complicated process, it's better to break down the process into the smallest possible parts and build tests for those. These small tests can then be combined in complex workflows using the right automation solution. This way, small changes won't break down the entire script.

Another aspect is building a keyword library. Many test automation solutions have inbuilt libraries. Building test cases using these keywords becomes easier, even for people with no experience with test automation.

Lastly, continuous testing with test automation is not possible with the inputs of just one department. The entire organization needs to come together and communicate goals clearly to make the most out of test automation.

Why choosing Avo Assure as the test automation partner is the right choice?

Test automation has become one of the most sought-after technological solutions today. While the market is filled with products that claim to work best, most require extensive coding knowledge. This forces companies to either outsource their testing to an external agency or invest in building an in-house test automation team from scratch. Both these options can cost a lot.

This is where Avo Assure comes in. It's a robust no-code, heterogeneous test automation solution with a pre-built keyword library apart from additional features such as easy reporting, smart scheduling, and accessibility testing. A potent combination of 1400 keywords, intelligent reporting, and easy-to-use UI enable even non-tech people to build and deploy test cases.

Here are some other ways in which Avo Assure makes testing easier for you

- Saves time and money by allowing you to run tests 24/7.
- Enables you to test any application of any size end-to-end with just a few clicks in very little time.
- Ensures that you get the expected test results every time with no room for error.
- Allows you to rapidly implement continuous testing and shift left testing, both crucial
- components of DevOps and agile.
- Allows you to reuse test scripts, reducing time and effort during regression testing.
- Creates a shorter feedback loop, directly impacting development quality and customer experience.
- Improves team morale by allowing testers to automate mundane tasks and instead refocus their energy towards
- more productive tasks.
- Builds and protects your test database effectively.
- Enables seamless reporting with screenshots and videos making it easier to catch bugs.
- Makes it easier to scale testing in a matter of minutes by simply adding test executors.

Avo Assure doesn't just promise a continuous, heterogeneous, and end-to-end testing solution but also provides significant cost benefits and time savings, freeing up your personnel to focus on more creative and critical tasks.

SAP® Certified
Integration with SAP NetWeaver®

Available on
SAP Store

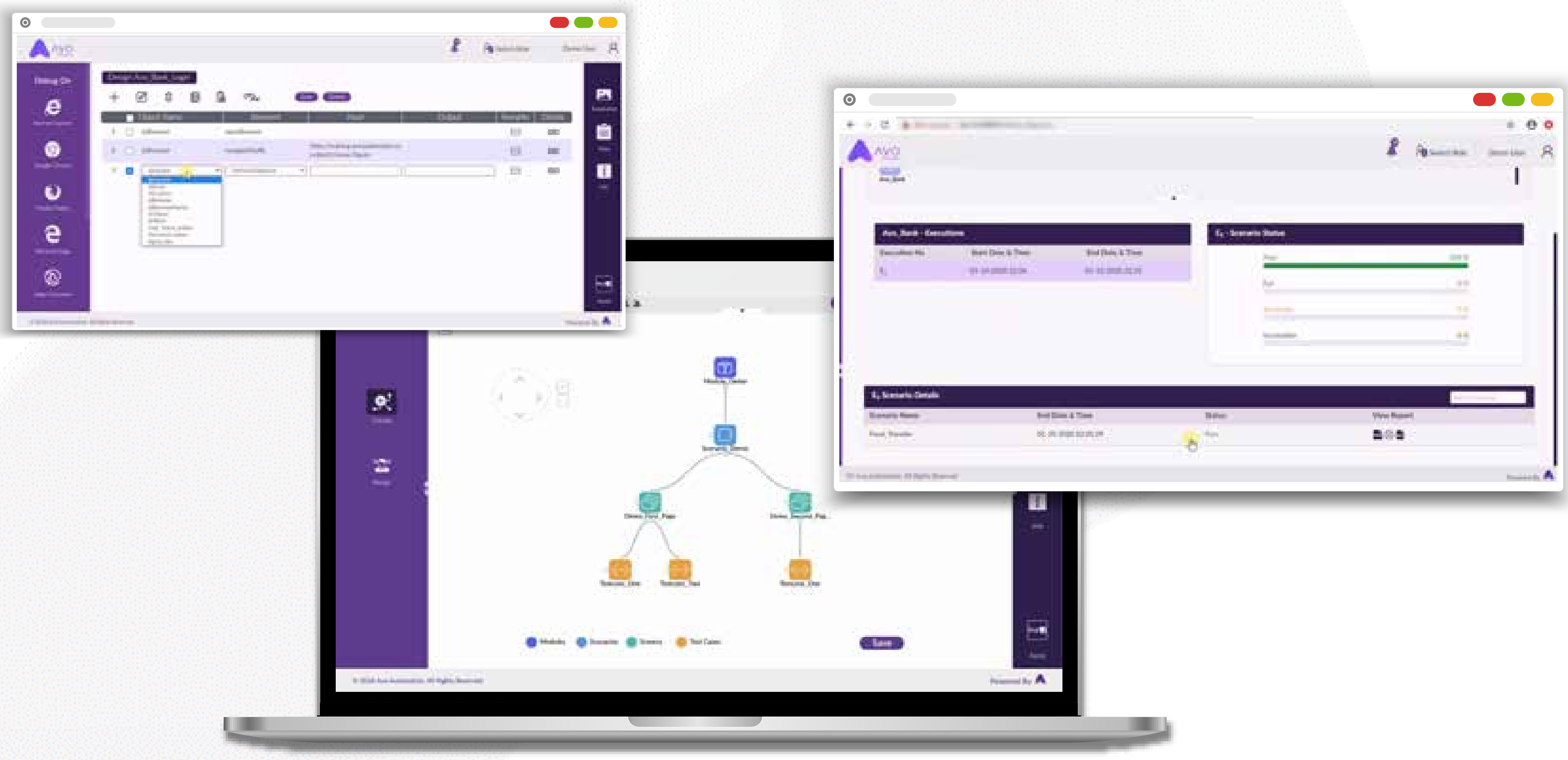
SAP® Certified
Integration with SAP S/4HANA®

High Performer SUMMER 2021

Users Love Us

High Performer Enterprise FALL 2021

High Performer FALL 2021



If you want to know how Avo Assure can help your enterprise out, then sign up for a **demo** today.

Schedule a demo

You could also opt for a free 14-day trial to test a scenario of your choice. **Click here.**