Table of Contents

**Test Automation**2

Pros of Automation Testing2

Cons of test automation3

**Toolsets for test automation** 3

Automation Tool select criteria4

What are the considerations or criteria to indicate automation tests should be used?4

Return on Efficiency ………………………………………………………………………………………………………………………..5

**Testing levels and types through Test Automation**…………………………………………………………………..5

**Robust Framework**……………………………………………………………………………………………………………………7

**Basic test Automation Framework**……………………………………………………………………………………………8

**Automation Testing Document**

**Test Automation:**

Automation testing is the process of executing the testcases through the scripts written with the automation specialized tools against the software application. Testcases are executed automatically without manual intervention to improve the efficiency, accuracy of the testcases.

It allows the Tester to perform repetitive tasks without any manual intervention. Automated tests are used to automate tasks that are hard to do manually and monotonous. Automation tests, which employ predefined processes to evaluate the software, can be conducted at any time of day. They use scripts to test software. They take input as test data and based on script compare the expected test result to the actual test result. They generate detailed tests reports. The test suit can be recorded and re-created when needed. The purpose of automated tests is to reduce human intervention/execution not to disregard the manual execution.

**Pros of Automation Testing**

There is a need to perform certain test multiple times and such testing become monotonous and there is possible of not covering some test with different data set which may further introduce the defect and failure. In such a scenario Automation help and give relief from repeated task. With this below are the key reasons why Automation is required

1. **Quality Assurance:** Manual testing is a tedious task that can be boring and at the same time error-prone. Thus, using automation testing improves the quality of the software under test as more test coverage can be achieved.
2. **Error free Software:** Automation testing is more competent for detecting bugs as compare to manual testing.
3. **No/Less Human Intervention:** Manual testing requires huge human effort in comparison to automation testing which requires no human intervention and the test cases can be executed in unattended mode
4. **To Increased test coverage:**  It raise test coverage in comparison to manual testing where it is not possible to achieve 100% test coverage.
5. **Frequent tests**: Automation testing means that the testing can be done frequently which improving the whole quality of the software
6. **Reliability:** Since it is performed by third-party tools and/or scripts, therefore it is more reliable.

**Cons of test automation**

1. **High initial cost**: In the initial stage requires a lot of time and money to be invested. It requires a lot of effort for selecting the tool and designing customized software. So if the feature and relevant test are having limited use and deployment the we should think on ROI and then plan for Automation
2. **False positives and negatives:** Automation tests may sometimes fail and reflect that there is some issue in the system but there is no issue present and in some cases, it may generate false negatives if tests are designed to verify that some functionality exists and not to verify that it works as expected.
3. **100% test automation is Myth**: In practical real situations not all test cases can be automated some test cases require human intervention for careful observation. There is always a human factor, i.e., it can’t test everything like humans(design, usability, etc.). and it is not possible to automate all test types: It is not possible to automate tests that verify the user-friendliness of the system. Similarly, if we talk about the graphics or audio files, even their testing cannot be automated as automated tests typically use textual descriptions to verify the output.
4. **Programming knowledge is required**: Every automation testing tool requires any one of the programming languages to write test scripts, it is compulsory to have programming knowledge for automation testing.

**Toolsets for test automation**

There are some popular tools which enable the Automation testing and need knowledge of programming knowledge too.

1. **Selenium:**Seleniumis an automated testing tool that is used for Regression testing and provides a playback and recording facility. It can be used with frameworks like JUnit and Test NG. It provides a single interface and lets users write test cases in languages like Ruby, Java, Python, etc.
2. **QTP:**Quick Test Professional (QTP) is an automated functional testing tool to test both web and desktop applications. It is based on the VB scripting language and it provides functional and regression test automation for software applications.
3. **Sikuli:**It is a GUI-based test automation tool that is used for interacting with elements of web pages. It is used to search and automate graphical user interfaces using screenshots.
4. **Appium:**Apium is an open-source test automation framework that allows QAs to conduct automated app testing on different platforms like iOS, Android, and Windows SDK.
5. **Jmeter:**Apache JMeter is an open-source Java application that is used to load test the functional behavior of the application and measure the performance.
6. **Tosca** : allows users to easily capture test cases and data from applications and create parametric test cases, making the creation process faster. Once created, these cases can be automatically tested across various platforms for efficient testing results.
7. **Test complete**: Test Complete offers a range of powerful features to help you scale your automated UI testing efforts and maximize test coverage. This is s a script-less tool which does not require mandatory programming skills to start with. It is very easy to use and helps in creating robust scripts, but cost is higher.

**Note**: There are plenty of tools in market but which need to be chosen depend on the Organization (cost, policy , licenses), available skill set and nature of project ( data sensitivity , criticality etc)

**Automation Tool select criteria:**

* **Ease of usage:**Some tools have a unreasonable learning curve, those may involve users to learn a completely new scripting/programming language to create test cases and some may require users to maintain a costly and large infrastructure/environment to run the test cases.
* **Support to multiple browsers:**Cross-browser testing is crucial for acceptance testing. Users must verify how easy it is to run the tests on distinct browsers that the application supports.
* **Flexibility:** single tool framework can’t support all types of testing, so it is recommended to wisely examine what all tool offers and compatibility with other tools
* **Easy to Analyze:**All tools don’t support the same sort of analysis. Some tools have a user-friendly dashboard feature that shows all the statistics of the test like which test failed and which test passed. other side, some, first request users to generate and download the test analysis report thus, not very user-friendly. It depends entirely on the tester, project requirement, and budget to decide which tool to use.
* **Cost of tool:** Some tools are free, open source and some are commercial tools but many other factors need to be considered before deciding whether to use free or paid tools. If a tool takes a lot of time to develop test cases and it is a business-critical process and carry sensitive data that can bring concern in future, then it is better to use a licensed tool that can produce test cases easily and at a quicker rate.
* **Availability of support:**Free tools mostly provide community support; however, commercial tools provide customer support, and training material like tutorials, videos, manuals and chatbots etc.

Nature of Application/Product**,** complexity, usage, business criticality and there are many other factors with respect to organization which need to be considered before selecting the appropriate tool.

**What are the considerations or criteria to indicate automation tests should be used?**

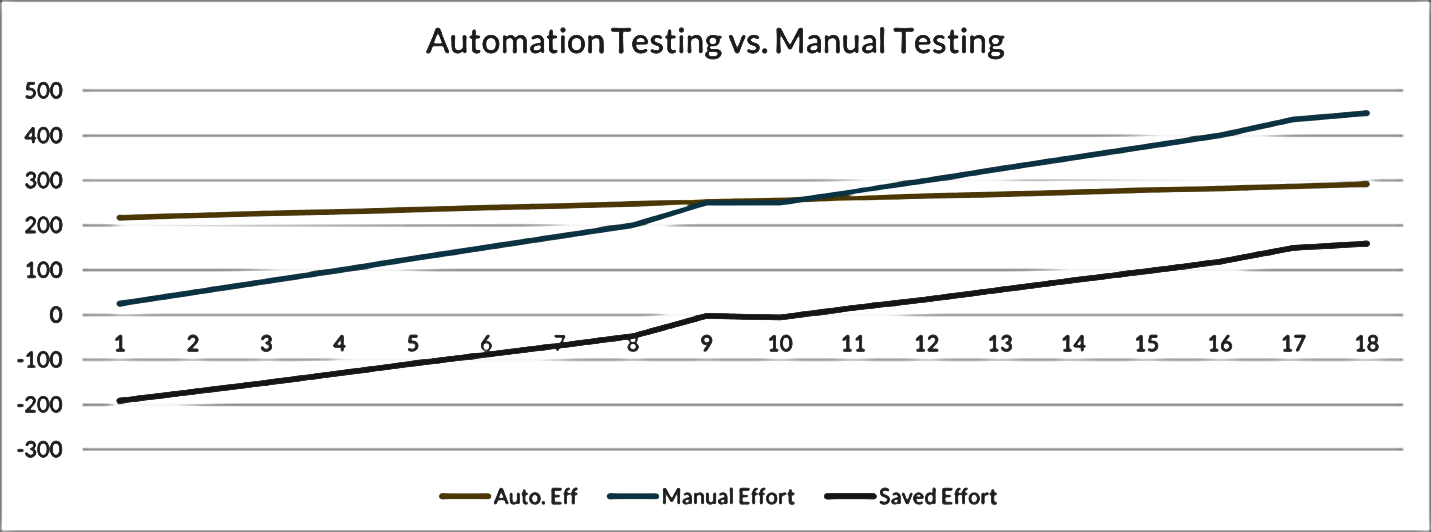
There are some basic criteria which help to decide when Automation needs to be triggered:

1. **Business critical tests:** Critical test cases should be automated to schedule to run regulary.
2. **Monotonous test**: Repeatable and monotonous tests should be automated to reduce the manual effort in future.
3. **Parameterized tests:** Multiple data sets with the same scenario should be automated.
4. **Determinant test**: Determinant test cases where it is easy for the computer to decide whether the test has failed or not can be automated.
5. **Tedious test**: Test cases that needs repeatedly doing the same action can be automated so that the computer can do the repetitive task as humans are very poor at performing the repetitive task with efficiency, which increases the chances of error.

Although, Test automation is a key component of quality assurance which improves the speed, reliability, and efficiency of testing and brings the Quality to application /Product. But it is required to measure the return on investment (ROI) of test automation, which is a key indicator to align the costs and efforts involved in implementing and maintaining a test automation approach and plan.

There are ways to measure ROI on test automation.

**Time saving**: -The basic calculation that shows amount of time can be saved by replacing manual testing with automated testing and more complex calculations based on efficiency and risk reduction.



**Cost /Effort Saving:-** It gives the amount of time /Effort saved by running a certain number of automated tests over a specific period instead of performing manual tests and calculate the cost over those effort :

**= {(Benefits- Cost) /Cost} 100**

**Return on Efficiency: it focus on investment gains over the period. In this case, the investment and cost that go into the ROI formula is measure as**

**Investment =** Automated test script development time + Automated test script execution time + Automated test analysis time + Automated test maintenance time + Manual test case execution time

**Gain** = Manual test execution time or analysis time Total number of test cases (automation + manual) Period of ROI / 8

**(above can be turned into Effort /$ value)**

**Testing levels and types through Test Automation**

1. Automation testing has various testing levels such as Unit testing (Modules are automated), Integrated testing(Integration of different modules are automated), Functional testing(Automating the functional aspects), Non Functional Testing( Automating the Performance, Security , Usability and Reliability Testing)

There are some test examples that can be considered as **good** candidate for an automated test:

* + Tests those are required to be run against every build/release of the application, such as the smoke test, sanity test, and regression test
  + Tests that utilize the same workflow but different data for its inputs for each test run data-driven and boundary tests.
  + Tests that need to gather multiple information during runtime, like SQL queries and low-level application attributes.
  + Tests that can be used for performance testing, like stress and load tests.
  + Tests that take a long time to perform may need to be run during breaks or overnight. Automating tests maximizes the use of time and no need to attend can be scheduled.
  + Some tests involve inputting large set of data.
  + Tests that need to run against multiple configurations and with different OS & Browser combinations.
  + Tests during which images must be captured to prove that the application behaved as expected.

The test which need to be avoided to Automate

* Usability or User experience tests where require a user to gauge how easy the app is to use
* Tests that need to be run one time, as a rule. If test is associate to a specific feature or requirement specific which will be having limited usage
* Test that needs to run ASAP on ad hoc and random basic which need exploratory and done based on domain or subject matter expertise/knowledge
* Tests without predictable results or know expected result. Where we need to deal with unknown and need analysis to produce pass-and-fail conditions
* Tests that cannot be 100% automated should not be automated at all — unless doing so will save considerable time.
* Test that don’t add value to quality of application
* Test that doesn't focus on the risk areas of your application but may increase cost
* **Common challenges or pitfalls surrounding test automation**

Test automation is a crucial aspect of software development, helping teams achieve faster feedback, improved efficiency, and enhanced product quality. However, implementing and execution of test automation is challenging. There are common pitfalls

1. **Lack of Planning and strategy** – Most of the test automation fails is simply because of not considering the time and effort required and wrong estimation, which cause the failure and bring halt in Automation.
2. **Inadequate Tool Selection** – lack of understanding and analysis towards fitment and technical skills required to run tools and attractive feature like (Free/Open source, click ,record and play , and hypes created by AI brings the failure for Automation
3. **Lack of scalability –** with time elapsed, tests grows with complexity, requires broader test coverage , hence the framework sustainability goes away if framework and tests are not scalable
4. **Brittle and Poor test case design** – Repetitive steps, incomplete steps, inappropriate size of test poor test data /preconditions, Hard coded data, insufficient object identifiers, implicit wait or fixed wait, not correct clean-up and closure the system post system also cause the failure.
5. **Overlooking test maintenance-** Tests must be maintained and monitored to align with the functionality and feature to ensure the 100% coverage, which is not possible all the time. Lacking in getting desired test coverage curtails the perceived value, thus testers focus shifts and test automation suite gets fail.
6. **Missing Integrated into DevOps CI/CD** **and compatibility with other tools stacks** : while Automating the test cases goal with respect to DevOps /CI/CD goal is to increase the speed and reduce the cost of delivery while maintaining quality but not focusing on rubus integration and compatibility check with other tools breaks the Automation roadmap

**What are the guidelines when maintaining automated tests and tools?**

Below are some of the best practices for test automation that can be followed:

**Robust Framework:**

**Create Robust Framework considering below steps**:

* Establish a test automation plan, project needs to have with a strong strategy which is agreed by all stakeholders
* Choose a test automation approach considering all aspects like usage, cost, organizational compliance, Nature of project complexity etc.
* Obtain a testing tool from open source or a licensed one.
* Set up an automation environment with required infra and CI/CD integrations.
* Design test automation scripts.
* Execute test automation.
* Implement monitoring mechanism (Result, Analysis, and Reusability.

**Plan self-contained test cases**: Clearly define the testcases so that user can easily understand .

**Plan the order to execute tests**: Planning the order for the execution will ease the process.

**Use tools with automatic scheduling**: If possible, use tools that can schedule testing automatically according to a schedule.

**Set up an alarm for test failure**: If possible, select a tool that can raise an alarm when a test failure occurs. Then a decision needs to be made whether to continue with the test or abort it.

**Reassess test plans as the app develops and changes:** It is important to continuously reassess the test plan as there is no point in wasting resources in testing the legacy features in the application under test.

**A test automation framework** is defined as a substantial or conceptual structure; created to provide support which can be enhanced in future. Test Automation framework is not a single process or tool. It is a collection of tool and process working together to automate the manual process. Automation Framework uses various libraries, test data and various reusable modules to automate the manual process to save the time of the tester. Testing Framework is a set of rules which is used for creating and designing test cases.

**Basic test Automation Framework:**

**Test Goal**

**Business Logic**

**Configurations**

**Resources**

**Objects**

**Pages**

**Utility   
Logger  
Handlers**

**Reporting**

**Test Runner**   
Web Driver   
Web Services

**Common API/Functions**

**Commonly Used Test Automation Frameworks:**

Linear Automation Framework -The linear Automation framework is commonly used in the testing of small applications. This framework is also called as a Record and playback framework only benefit is ros: There is no need to write custom code, so expertise in test automation is not necessary but its is not sustainable and scalable as data comes as hard coded reusability also get vanished as we can’t alter data

Modular Driven Framework - In this Framework, the tester can create test scripts module wise by breaking down the whole application into smaller modules as per the client requirements and create test scripts individually. : Modular driven framework ensures the division of scripts that leads to easier maintenance and scalability. You can write independent test scripts, although it reuires additional time in analyzing the test cases and identifying reusable flows.

Behavior Driven Development Framework - Behavior Driven Development framework is to create a platform, which allows every person, like Developers, Testers, business analyst, etc. , to participate actively. It also increases collaboration between the tester and the developers on your project. It use non-technical, common English language to create test specifications on this behavior-driven testing, but work with this framework and write scripts, sufficient technical skills as well as prior experience in Test driven development is required.

Data-driven Testing Framework -Generally, Test Data is read from the external files like Excel Files, Text Files, CSV Files, ODBC Sources, DAO Objects and they are loaded into the variables inside the Test Script. The data-driven framework allows us to create test automation scripts by passing different sets of test data. It reduces the number of scripts required. Hence, multiple scenarios can be tested in less code, on the downside it needs highly experienced software testers who should be proficient in various programming languages to completely utilize the design of this framework.

The Keyword-Driven Testing Framework - The keyword-Driven Testing framework is also known as table-driven testing. This framework is suitable only for small projects or applications. The automation test scripts performed are based on the keywords specified in the excel sheet of the project. A single keyword can be used across multiple test scripts, which gives heavy essence of reusability The initial cost of setting up such framework is time-consuming & complex which induce the high cost

The Hybrid test Automation Framework - Hybrid Framework is used to combine the benefits of keyword-driven and Data-Driven frameworks. This type leverages the advantages of all kinds of related frameworks. Tests are fully scripted in a Hybrid Testing Framework thus increasing the automation effort.

All these test automation frameworks can be effectively used to handle the code in a methodical way, which can be reviewed by a third person easily. Selection among these depends as per your project requirements, team expertise, time as well as budget. Test automation frameworks enhance productivity by adopting the framework, maximum test coverage can be achieved consistently

Different Framework and Respective tools:

|  |  |  |
| --- | --- | --- |
| **Framework** | **Key feature** | **Standard Tools** |
| Linear Automation Framework | Step-by-step actions, Record and playback | Selenium IDE , Katalon IDE, QTP |
| Modular Driven Framework | Enables modularity, reusability and robustness and independent test | Selenium WebDriver |
| Behavior Driven Development Framework |  | Selenium with cucumber, Playwright , test complete |
| Data-driven Testing Framework | drive test cases and suites from an external data feed | Tosca, TestNG, JUnit, or libraries like Apache POI or CSV Reader |
| Keyword-Driven Testing Framework | based on keywords that form the basis of the functionality, take in parameters, and throw the relevant output | Test complete, Selenium with TestNG or Junit, Ranorex |
| Hybrid test Automation Framework | Leverage both Data-Driven and Keyword Driven frameworks | Cypress, Selenium, Playwright |