

# **SOFTWARE TESTING**

By

**JUTTHADA DHANA LAXMI**

**ROLL No. 206E1A0512**

**&**

**TALITARA PAVANI**

**ROLL No. 206E1A0532**

A REPORT SUBMITTED TO

GONNA INSTITUTE OF INFORMATION TECHNOLOGY & SCIENCE,

VISAKHAPATNAM

FOR

**INTERNSHIP**



**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**

**GONNA INSTITUTE OF INFORMATION TECHNOLOGY & SCIENCE,**

**VISAKHAPATNAM-53 (INDIA)**



**Department of Computer Science Engineering**

**GONNA INSTITUTE OF INFORMATION  
TECHNOLOGY & SCIENCE, Aganampudi**

**Visakhapatnam – 53**

---

## **CANDIDATE DECLARATION**

I hereby declare that the work which is being presented in this report entitled “**SOFTWARE TESTING**” is a genuine work carried out by me i.e. **JUTTHADA DHANA LAXMI (206E1A0512)** and **TALITARA PAVANI (206E1A0532)** in the Department of Computer Science Engineering, Gonna Institute of Information Technology & Science, Aganampudi, Visakhapatnam, affiliated to Jawaharlal Nehru Technological University, Gurajada Vijayanagaram during academic session 2022-2023.

**Date:**

**Place:** Visakhapatnam

**Jutthada Dhana Laxmi**

**Talitara Pavani**

## **ABSTRACT**

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

This report contains observations and recommendations based upon the internship. This report also includes position statements submitted to the internship and presentation slides. Presentation addressed software testing standards; best practices in testing; manual verification and automation verification.

**Keywords:-** Katalon, selenium, Software testing, manual testing, automation testing.

# Chapter 1

# Introduction

The term automated testing describes the use of computerized testing tools, for automating the software testing process. This chapter aims to explain the concept of software test automation, discuss the pros and cons of automated testing, brief the process of test automation, list different types of test automation, talk over its impediments, defining roles, responsibilities and exploring careers in test automation, while covering the comprehensive details of Selenium Web driver.

## Introduction and evolution of automated testing

The process of software development is lengthy and time consuming. It comprises of various stages such as Evaluation, Requirements, Analysis, Design, Implementation, Validation and Deployment. Validation testing when compared to the different stages of software development comes later in the process. This is the time when pressure of software delivery is higher than other tasks. Therefore, one of the reasons led to automate the testing process is to speed up the project. It is estimated that the testing phase can take up to the 50% of project development resources. So, this process is said be expensive. Moreover, many factors are accountable in crossing deadlines of successful software product delivery such as late modifications in requirements, and issues related to developers such as leave or changing of job and etc. It might happen that, in such cases the developers might miss the deadlines that may result in delay of project completion and furthermore may make the testing process even more deferred, hurried and expensive. Also, organizations may prefer to test the software systems more adequately, but in minimum time and resources. Therefore, many organizations are finding automating testing as more useful and are turning towards automated testing.

Automated testing is the way of testing software programs using automated tools. In other words, the process of automated testing is supported by computer-aided software engineering. When the testing is performed manually, the responsible person or team is able to plan design and run the tests. Software test automation represents the use of software instead of person or team to control the execution of tests, the comparison of actual outcomes to predicted outcomes, the setting up of test

preconditions, and other test control and test reporting functions. However, the development of tools for the purpose of automated testing is costly and timeconsuming job [1-7].

The automated tool is defined as program that check the presence of definite program attributes such as program syntax correctness, correct program control structure, appropriate program module interface, and testing completeness.

An automated test framework may be loosely defined as a set of abstract concepts, processes, procedures and environment in which automated tests will be designed, created and implemented. In addition, it includes the physical structures used for test creation and implementation, as well as the logical interactions among those components.

There are many situations when automated testing is recommended:

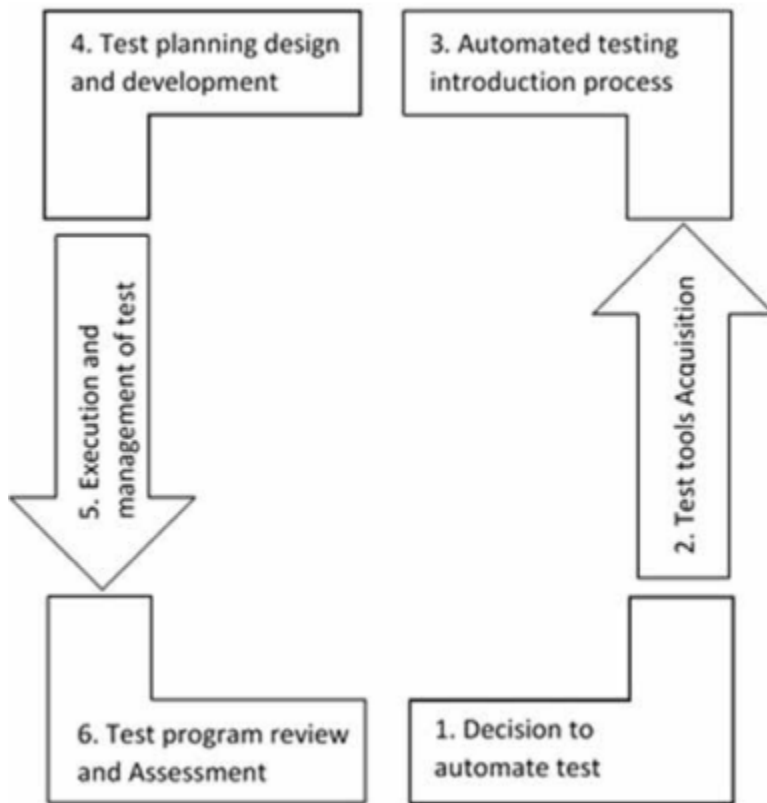
- Automated testing is efficient to perform, as compared to manual testing.
- The automated testing is suitable for complex systems. The testing is carried out before placing the software systems for the final deployment.
- If there are many chances of regression testing (repetitive testing) then automated testing is effortlessly supporting the repetitions. It will not only helpful for carrying out the testing but also helpful in generating the caparison log files.
- The software size is big and conducting manual testing have complexity then it is recommended to conduct the automated testing.
- Many black box test's such as stress test, load tests, and durability test can only be conducted automatically because it required many testing resources. Simulated testing is only solution for such type of tests.

In following few situations, automated testing is not recommended:

- High cost is one of weakness of automated testing.
- The development of the tools for automated testing is extremely high and require many resources. Therefore, automated testing is only recommended in such situations when there is a real need.

## Process of Automated Testing

Planning of the automated testing in early stages of software development life-cycle is important for the success of the test. Therefore, test engineers should be included from the phase of business analysis, requirement activities and further, the team needs be involved in analysis, and design reviews activities. Such early involvement allows the test team to gain good understandings of customers' requirements and it helps in developing an appropriate test environment and producing more detailed test designs. Usually the test strategy should be defined at the stage of requirements specifications phase in software development lifecycle. The automated tool that support the testing process will be helpful to produce functional requirements that will help later in automated testing and will reduce the efforts and cost of testing. If the decision of test automation made early in the software development cycle then the product design and coding standard can provide a feasible environment for the test automation [8-11]. Figure 8.1 presents Automated Test Life-Cycle Methodology (ATLM) that consists of six different phases.



**Figure 1.1:** *Automated Test Life-Cycle Methodology (ATLM) (Elfriede Dustin, 1999)*

## Test tools acquisition

The selection decision of testing tool should be making smartly that will fit with the organization's system engineering environment. For this purpose, the test engineer is required to study the organization-engineering environment. While requesting for the automated testing tool the test engineer should be confident that the tool should fit with the various operating systems, platforms, programming languages being used by the organization. Surveying is one of the recommended techniques to understand the organization's requirements. The survey could help to identify the usage of automated tool within or outside the organizations, department or groups, most or least important functions and tests, the main purpose of tool, and



portability of the tool. The survey could help the test engineer to correctly identify such tool for the organization that will fit with the engineering environment of the organization. Further, the test engineer needs to review the systems and software architecture of the organization to identify most commonly used for the systems development. Ideally, identified tool should be utilized in every phase of development life cycle:

- Business analysis phase tools
- Requirement definition phase tools
- Analysis and design phase tools
- Implementation and development phase tools
- Testing phase tools
- Software quality metrics tools
- Other testing life-cycle support tools

It is important not to leave the selection of automated test up to the criteria of one project, because the test team the test team will take the advantage of one test tool in various projects within the organization. The level of software system quality is important to consider for the test manager while selecting the test tool. A few software systems possess high quality and in the case of failure, the organization could bear with a great loss in terms of money or lives. The test manager should also identify the parts of software system that require intensive testing and possess high priority within the system. It is also important for the test engineer to limit the automated testing tool on available budget. In addition, identifying the test tool requirement is important, as regression test, stress test, or usability testing is important within the organization. The organization should be willing to provide the resources so that the test engineer could effectively introduce the test tool with the organization.

Based on search tool requirement the test team should prepare test tool specification and evaluation form. The literature survey should be made through articles, technical reviews, surveys, newsgroups, forums and vendors' advertisers to explore success stories of tools and their technical specifications. After getting collection of the automated testing tools, the list can be narrowed down by eliminating such tools that do not meet minimum requirements, furthermore, select such tools that meet the minimum requirements of organization. For this purpose, the requirements criteria could be made in the form of list and priority will be assigned on 1-5 Likert scale. The tool that score the highest will be a candidate tool for selection. Based on selection process the test engineer should be able to get the idea about the automated test tool suitable for the organization technical environment. Furthermore, the evaluation of the tool can be done before utilizing it for the software testing purpose. Now, the vendor of the tool should be contacted to make demonstration of the tool within the organization. The test engineer can ask for trial version of the test tool for the initial evaluation purpose. The evaluation report should be generated to compare the rating of the automated testing tool with previous research-based rating of the tool. If there is a significant difference between the both ratings, the test engineer is required to reconsider the selection of the tool of the tool. If the test engineer is satisfied with the performance of the tool then the test engineer should go for purchase by verifying the license agreement according to system operational requirement.

## **Automated testing introduction process**

In order to decide the test process for the organization the test team should become familiar with the quality and process improvement standards. The test process must be documented for effective execution [18,19, 21]. In order to introduce the automated testing, the test process should identify these requirements:

- Testing goals are identified and objectives should be defined.
- Test strategies are developed.

- The tool required to implement the strategies should be made available.

# Chapter 2

## **Test planning design and development**

It is important for test team to be involved in development process in order to corporate automated testing in the application. The design team is required to develop an effective test plan documentation that include: the team roles and responsibilities, project testing schedule, test designing activities, test environment requirements and preparation, identification of testing risks, and testing level of acceptance. The appendix of testing document contains testing procedures, naming conventions, and requirements to testing procedure compatibility matrix [10, 12, 13]. While starting test automated test planning the following criteria is helpful to follow:

- In the beginning of process, it is important to investigate, whether, it is beneficial to adapt automated testing for the type of project, available resources, technical environment, platform, and user environment.
- In order to adapt automated testing, it is important to evaluate and select a suitable automated testing tool to support test efforts. Because of selecting one or more tools for automated testing, there should be a document stating the reasons for opting the tools.
- The testing goals and objectives need to be defined and analyzed. The result of analysis will be refinement of test objectives and goals.
- The test team should verify the test tool suitability for the project requirements, the project environment, and the schedule.
- The test team must identify the scope of automated testing. Furthermore, the test team should identify various tasks of automated testing and schedule them according to the priorities.
- It is also important to document the expertise of the test team members such as their qualifications, experience, and desired knowledge. Correspondingly, the roles and responsibilities of the test team needed to be documented in the test plan.

In the test-planning phase, the focus is on test requirements definitions and identifications of test-required documentation, planning for test environment, and development of test plan document. The test program scope must be defined in the test-planning phase considering the test efforts in terms of personnel, person hours, and test schedule. The test plan includes the program assumptions, prerequisites, and risks involve need to be documented. It is fact that not everything could be tested, therefore, it is required to identify high-risk functions and critical success functions. The requirements traceability matrix helps to provide the tester with the ability of keep track of test coverage of requirements. The test plan also involves the hardware, software, test environment need to support the testing process. Overall, design approach of testing needs documentation in the test plan. Planning document, include identification of procedure for test scripting. Initially, the test schedule builds that corresponding to the development plan. The development of test planning is not a simple task, as many factors need to be identified in this phase so it require considerable amount of effort. After any revisions and refinement when test plan will be ready, it will work like a guiding instrument for the test team for the testing activities in the project.

In order to design the testing, process the test requirements and design need to apply:

- Just same as the requirements gathering phase in the software development, the test requirements are required to gather and define and must be specified before the design.
- There is limited scope of the test requirements and bounded by the system description and system requirements definition.
- There are two approaches i.e. black box and white box that can be applied for the testing purpose. The black box techniques cover more details with the functionality of the systems. The white box techniques are concern with the internal structure of the system. The appropriate technique could be adapted for testing purposes.

The test procedures will be designed after test requirements are gathered and analyzed. The test procedure consists of group of test procedure and naming convention. Then the procedures can categories as automated or manual type of testing. The test design is needed throughout the software development cycle phases. In the requirements gathering phase, the test requirements are already extracted, now in the design phase it is required to develop test procedure against each requirement. The well-defined test procedures, has availability to uncover the errors in every phase of development lifecycle. The test design should comply with naming conventions, design standards, and the templates.

In order to develop detailed test procedure first, the test team should review the test procedures at system level and identify those procedures that require detailed design. Secondly, test team will list all the test procedures that are selected for detailed design. Lastly, the test team will create the detailed design document (see Table 8.1) for the all procedure in the list. Next step is to gather the test data foreach test procedure selected for detailed design. After completion of test procedure design, now the test team is ready for test development.

Section	Description
Introduction	The purpose of document and criteria of selectin test procedures for detailed design. g
Test procedure list	List the test procedures selected for the detailed design and thereason of requiring detailed design for them.
Detailed design	The procedures can be arranged in order and provide the detailed design for each procedure.

Summary	Summarize key points and notes regarding design and development of design.
---------	--

*Table 2.1: Detailed design document outline*

## Execution and management of test

After completing planning, designing, and development process of test procedure, the test team is ready for test execution. The execution of automated test can be performed using automated test tool or test management tools. Robot framework [14] is generic framework that is used to process test data, execute test cases and generate the logs. With the help of this framework the support is provided for both internal and external libraries that automate the test cases for the front-end GUI web application (Selenium), backend testing (SSH, Database), Windows application (AutoIt), mobile application (Android, IOS), and so on. Similarly, Testrail [15] is another example of modern test case management software that is use to managetest cases, plans and execute them.

The primary input of each test phase is dependent of the test procedures. The output of each test phase is according to the defined acceptance criteria as defined in the test plan. The software problem reports are documented throughout the test phases, and problems fixes are documented. After completing the test execution, the test team will report the errors to the developers, later the automated defect correctionis performed using defect-tracking procedure. The problems are reported according to the common classification of defect priority levels such as fatal, high priority, medium priority, and low priority.

1. **Fatal:** The error cause interruption in the operation of application.
2. **High priority:** The application is operational with a significant problem.
3. **Medium priority:** The error occurred with little impact on the application.
4. **Low priority:** There is no impact of error in the operation of application.



## **Test program review and assessment**

Throughout the testing progress, the test team collects various test metrics. The test metrics provide the measurement of quality of test program and sizing as compared to actual labor-hours for the manual testing. The outcomes of test metrics conclude the suggested adjustments and improvements recommendations. The documentation activities form test team serves as guideline to follow the successful procedures. It is good idea for the test team to record lessons learned throughout the testing process. If the test team documents all activities that were performed well and correctly then these will be surely beneficial to use in the next testing activities [16].

# Chapter 3

# Types of Automated Testing

The test automation can be applied on both black box and white box testing types. The process of testing the executable program without referring the source code is called black box testing. White box testing deals with executing special test cases, in specific branches and paths in source code. Numerous types of test automations are available under the black box and white box testing labels. Few types of test are not feasible to perform manually. Table 8.2 show the classification of automated testing types according to the black box and the white box testing.

Automated testing types	Testing types
Functional tests	Black box
Code auditing	White box

*Table 3.1: Types of automated tests*

# Functional Test

The functional tests have many responsibilities, as it is kind of black box testing and main aim of these tests is to make sure that the system meet the customers' requirements. Furthermore, the functional tests are performed to ensure that the goals and metrics have been met, such as the systems performance metrics. The functional test, are often replacements of manual black box testing. The test cases are executed that produces the documentations in the form of graphs and results summaries and the tools provide the ability to produce statistical results on demand. Various kinds of tests fall under the umbrella of functional test.

**Accessibility Test:** Accessibility tests are formed, to test that how easily the users with disabilities can use the system. Many standards are available to follow minimum accessibility requirements in systems.

**Ad hoc Test:** In ad-hoc testing, the design is not formally done because the test is intended to run only once until the defect is discovered. There is not documentation required in such testing. These tests are considered exploratory tests.

**Acceptance Test:** The acceptance tests are intended to check the functionality that if the system is functioning exactly like how it is supposed to work. The acceptance is created from the customers' requirements and results in acceptance or rejection of the application.

**Capacity Test:** This type of testing is used to determine the maximum number of resources that contained inside the system under the test. In other words, determining that the application under test can handle how many users until its performance becomes unacceptable.

**Deployment Test:** Usually, deployment testing is done on virtual machines to test the functionality of applications on various platforms. Therefore, it is checked if the software is ready to deploy.

**Graphical User Interface Software Test:** The graphical user interface testing can be both automated and manual. The testing is performed to evaluate the GUI specifications are satisfying the users' requirements.

**Load Test:** The practice of modeling the expected usage of the application by generating a simulated multiple user concurrently is called load testing. There are various factors such as memory utilization, hardware capacity utilization, throughput and so on, considered to check the system response under maximum load conditions. Any unreasonable response of the system is corrected, so that application behaves in a better manner in production phase. Load tests are performed under maximal load conditions, therefore, it is impractical to develop such conditions manually, so computer generated simulations closer to real life load conditions, are required for testing purpose. The virtual users and virtual events are generated in computerized simulations to execute in hardware and communication environment defined by test planner. These virtual users and virtual events emulate the real user and real event's behavior and computerized simulations produce such outputs closer to captured from real life users and real-life events. These outputs are used as input of desired system test and the testing results help to finalize the hardware and communication configurations of final product [1-10].

There are many events, when despite systems are successfully passing the correctness test severely failed and cause huge damage. Often, this type of failure occurs in huge information systems when the system is supposed to serve a large number of users in real time firmware systems handling simultaneous events. Therefore, load testing is recommended to avoid such accidents.

**Performance Test:** Performance tests are performed to measure the response of the system when placed under the certain load. Therefore, the load testing and performance testing are interrelated to each other. Performance test is another name of load testing.

**Recovery Test:** System recovery tests investigate how well system recovers from

system crash and hardware problem.

**Regression Test:** Regression tests are regarded as re-testing of software systems. The automated regression tests are performed to verify that a satisfactory method of errors correction is performed by the developers. Furthermore, often it is possible that while correcting found errors the developers unintentionally introduce new errors to the application, so, regression tests are useful to cope with such situations and new errors are found in regression tests. In regression tests, once the test cases are designed and stored in database, later, this database will be used to run the tests with minimum efforts. One of the automated test tools called output comparator is a great help at regression test stage. As the output of successive tests and the results of functional testing tools will help the testers to prepare the improved analysis of regression test and help the developers to discover the causes of errors that detected in these stages. It is common to perform regression tests three to four times before they reach the satisfactory level of quality.

**Sanity Test:** Sanity tests are done to decide whether to continue or stop with more tests, by quickly checking the validity of function.

**Scalability Test:** This is an automated test to check if the website will continue working in the same manner if the users, traffic, and amount of data will increase anytime.

**Security Test:** Both manual and automated testing methods are used to ensure the protection of data and functionality under the test.

**Smoke Test:** The smoke tests are first quickly executed to test if the application will not catastrophically fail. If application doesn't unexpectedly fail, then more in-depth testing can be performed on the system. The smoke tests are often helpful in flushing the errors.

**Stress Test:** Stress testing is another name of performance testing. The stability of software application is tested by running simulated test data according to the

designed test cases.

**Usability Test:** The usability tests are performed to evaluate the users' performance on the system. Few available tools support the usability evaluation. Automated usability software is capable of capturing all of the user interactions with the system with any user interface device[4]. The usability testing automation tools will be able to:

- Monitor the users' interaction with multiple applications.
- Collect the logging data.
- Help access.
- Generating the usability results and reports, and so on.

**Volume Test:** The volume testing is concerned with usage of certain amount of data. Database files can be tested under the volume test. The volume testing is a type of performance test.

**Code Auditing:** The code auditing is type of white box testing that verifies the following:

- Does the code fulfil the requirements, guidelines, instructions, and procedures?
- Module size
- Level of loop nesting
- Level of subroutine nesting
- Prohibit constrictions such as GOTO
- Code auditor checks the coding style
- Naming conventions for variables, classes, file and routines so on.
- Unreachable routines and lines of codes
- The code auditors, confirms the documentations style according to the standards and guidelines.

# Chapter 4



## Automated Testing versus Manual Testing and their Pros-Cons

While comparing automated tests with manual tests, both types of testing have their own pros and cons. In few situations, it is appropriate to perform manual testing such as no time pressure, low budget, and small sized systems with less complexity, and so on. However, in many situations only automated testing is recommended such as the test requires huge datasets or unavailability of enough work forces, and so on [17]. There are many factors to be considered while selecting the automated testing over the manual testing.

### *Regression Testing*

If the software test involves regression tests then it is good idea to opt for automated testing over the manual testing because re-testing is time-consuming and performing regression tests using manual testing can be hectic task for the testing team. The automated testing results in better performance of complete regression tests.

### *Test accuracy and completeness*

The automated testing has the ability to perform all test procedures and test all cases up to full extent. The automated testing supports the execution of test case in complete and correct manner. The manual testing is much dependent on the testing team and testers, so, often it is suffered from low concentration and tiredness of the person.

### *Productivity of testing overtime*

It is found that the productivity of automated testing was doubled over the time after reusing the test library containing the basic routines and test engineers are advancing on the learning curve [13]. It is also effective to reuse test cases in the case of manual testing; however, performance is not comparable.

## *The formatting and completeness of test reports and logs*

The automated testing tools can produce more comprehensive and complete error reports than the one produced by tester in manual testing. The results and data can clearly be labeled, visualized, and presented in readable form. Naturally, during the test execution, the data and test results can be obtained and stored in the database, and it is easy to generate the comprehensive report from the available data using an automated tool.

## *Duration of testing*

The duration of automated testing is significantly less, than the duration of manual testing. Specially while testing the complex modules, and systems. The preparation phase of automated testing takes longer; however, the later phases compensate the time limitations.

## *Manpower Resources*

The automated testing require less workforce resources as compared to the manual testing. Therefore, opting for the automated testing is good idea in case of shortage of testing team.

## *Cost and budget*

The automated testing is expensive in terms of cost as compared to manual testing. As there is need to buy testing tool for automated testing purpose which is costly. However, the automated testing can be considered return on investment for its thorough identification of errors that reduces the chances of project failure. In case of testing, the complex and big modules, it is good idea to opt for automated testing, as compared to hiring huge number testing team members.

## *Considerable testing areas left uncover*

Currently, automated testing is limited in terms of tools available having ability to test everything. In such case, manual testing is required to cover up those modules that are not supported with the automated testing tools. There are few tests such as load tests, stress test, and durability test that cannot be performed manually, so

automated testing is suitable for such collection of tests.

## Impediments in software test automation

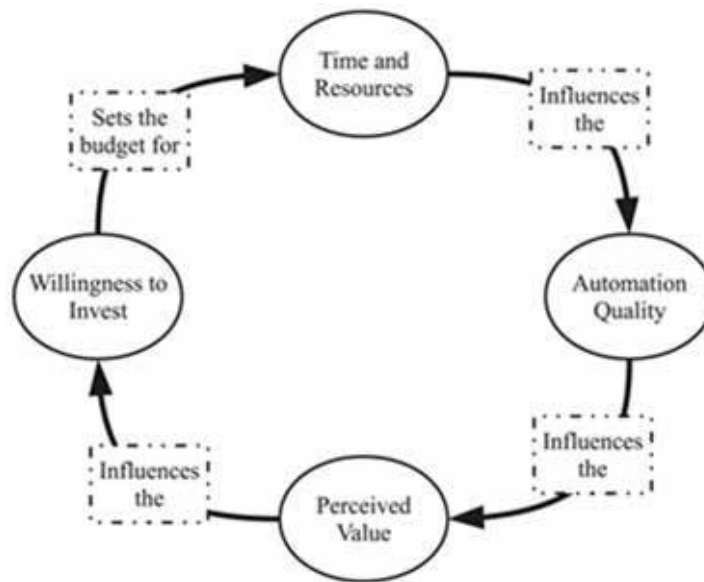
This section discusses the various types of impediments that can occur during software test automation.

Impediments are defined as *anything that prevents a team member from performing works as efficiently as possible*. It is significant to identify, prevent, and solve underlying conditions that can result in a test automation impediment. And is, essential to reduce their negative impact and solve them; in case if software testing is suffering from any unexpected hurdles, to mitigate the chances of product failures that may surface because of inefficient software test automation. Because of the fact, that test automation eliminates the possibility of human errors and saves time. Inversely, if the test automation does not perform well; due to any unidentified or unpredicted impediment, then it can affect badly on the software product. Hence, such conditions that prevent an automated test to execute efficiently must be identified and corrected timely.

With the iterative nature and the high requirement of performing test, it is evident that testing consumes most of the software development time. As stated, 30% - 80% of development time besides, occupying most of the release time is engaged by testing, by involving expensive resources such as humans and machines. Automating a test helps to improve the process with high efficiency with less cost and less resources. Continuous integration, Test-driven development and automated acceptance testing makes it a routine activity in agile software development.

Considering the fact that implementing test automation is highly looked-for; They have to be carefully designed before time and although if they are well prepared, still chances exist that they may fail due to the way they would be carried.

Figure 8.2 shows different factors influencing the test automation quality.



*Figure 4.1: Feedback between connected phenomena [3]*

Organizations not finding automation as useful as manual testing prefer to invest less amount of time and people which results in Low-quality automation. Therefore, knowledge about such limitation in both aspects i.e. Technical and Organizational are essential to deal with and eliminate them. They are further given as:

**Technical Impediments:** Impediments that may be fall due to insufficient test system and **system under test (SUT)** parameters, are termed as Technical Impediments. Such as Environment Configuration, Inadequate Development Practices, Quality Issues, Usability, IT Environment, and system under test. They are listed as follows:

### *Test System*

- Environment Configuration
  - Inadequate environment configuration management
  - Problems with configuration and setup

- Untested test environment
- Inadequate Development Practices
  - Missing specifications and Ad Hoc Development
  - Poor Test System Documentation
- Quality Issues
  - False Positives and Negatives
  - Fragile Test Scripts
  - Test Script Bugs and system bugs
  - Test System Instability
- Technical Limitations
  - Insufficient scalability and slow test system
  - Integration of several tools is necessary
  - Limitations in externally sourced tools and limited test tool functionality
  - Manual interventions needed
  - Problems with capture and replay tools
- Usability
  - Difficult analysis of test results
  - Low usability
- IT Environment
  - IT accesses
  - IT infrastructure
- System Under Test
  - Platform limitations and SUT complexity
  - SUT considered too small for automation
  - SUT Interfaces external systems
  - SUT quality and speed

- SUT testability

**Organizational Impediments:** Impediments that may befall due to lack of proper administration parameters such as behavior, planning, staffing, automation strategy, skill and etc. are termed as Organizational Impediments. They are further elaborated into the following effects:

- Behavioral Effects
  - Fear of redundancy and general negativity from automation
  - Lack of management support
  - Process deviations, considered too complex for automation
  - Testing is not viewed as a natural part of the work and shifting blame between development and test
  - Too high expectations on better testing
  - Expectations on high cost reduction and quick ROI
- Business and Planning
  - Automation is too expensive for small projects
  - Implementation, Maintenance, and Training Costs
- Staffing and Inadequate Strategies
  - Testing by non-testers
  - Inadequate automation and test strategy
  - Lack of Key Performance Indicator (KPI) for automation development
  - Communication
  - Change management of **System Under Test (SUT)** changes
  - Communication with tool suppliers
  - Roles and responsibilities
- Skills
  - Lack of testing and automation skills

- Lack of general development and programming skills
- Lack of application domain and steep learning skills

The test automation impediments described above are process specific, and may vary in existences from organization to organization. Most of the software testing teams suffer from organizational impediments if the company infrastructure fails to facilitate the state-of-the-art tools and domain specific trainings required to their software testers. On the other hand, companies have to bear product failures if the employees possess technical impediments. Hence, identifying, avoiding, and dealing with such hurdles in appropriate manner is highly essential to carry out successful automated test followed with fixed software product delivery in time.

# Chapter 5



## Review of automated tools

Software development practices change over time and so does the automation testing. In this era, efficient and easy tools are developed to carry out tasks with minimal human efforts. Automatically testing the web application is a good way to ensure that new versions of the application do not introduce bugs and regressions. Automation of the web application testing also allows the development team to make changes and refactor code with confidence. This enables them to keep pace with the trends in software application [20].

However, actually building automated tests for web applications is challenging, as the user interface of the application changes regularly. This leads to the incompatibilities between browsers as they need support from various servers or client's platform [16-18]. The following tools make it easier to build and execute automated tests for the web application.

### *Selenium*

Selenium is possibly the most popular open-source test automation framework for Web applications. It originated in 2000s and evolved over a decade. Selenium has been an automation framework of choice for Web automation testers, especially for those who possess advanced programming and scripting skills. Selenium is a core framework for other open-source test automation tools such as Katalon Studio, Watir, Protractor, and Robot Framework [23].

Selenium supports multiple system environments such as Windows, Mac, Linux, and browsers such as Chrome, Firefox, IE, and Headless browsers. The scripts can be written in various programming languages such as Java, Groovy, Python, C#, PHP, Ruby, and Perl.

While testers have flexibility with Selenium, and they can write complex and advanced test scripts to meet various levels of complexity, it requires advanced programming skills and effort to build automation frameworks and libraries for

specific testing needs.

## *Katalon Studio*

Katalon Studio is a powerful test automation solution for web application, mobile, and web services. This is built on top of the Selenium and Appium frameworks. The tool supports different levels of testing skill set. Non-programmers can find it easy to start an automation testing project for instance, using Object Spy to record test scripts. Whereas, programmers and advanced automation testers save time from building new libraries and maintaining the scripts.

Katalon Studio can be integrated with various processes like **Continuous Testing/Continuous Delivery(CI/CD)** processes and works well with popular tools in the Quality Assurance process including qTest, JIRA, Jenkins, and Git. It offers a unique feature called Katalon Analytics. This provides the users with comprehensive views for the test execution reports through dashboard including metrics, charts, and graphs.

## *Unified Functional Testing*

**Unified Functional Testing (UFT)** is a renowned commercial testing tool for functional testing. It provides a comprehensive feature set for API, web services, and GUI testing of desktop, web, and mobile applications across platforms. The tool has advanced image-based object recognition feature, reusable test components, and automated documentation.

UFT uses Visual Basic Scripting Edition to register testing processes and object control. UFT is integrated with Mercury Business Process Testing and Mercury Quality Center. The tool supports CI tools such as Jenkins.

## *Watir*

Watir is an open-source testing tool for web automation testing based on Ruby libraries. Watir supports cross browser testing including Firefox, Opera, headless browser, and Internet Explorer. It also supports data-driven testing, and integrates with BDD tools like RSpec, Cucumber, and Test/Unit.

## *IBM Rational Functional Tester*

IBM **Rational Functional Tester (RFT)** is a data-driven testing platform for functional and regression testing. It supports a wide range of application like .Net, Java, SAP, Flex, and Ajax. RFT uses Visual Basic .Net and Java as scripting languages. RFT has a unique feature called Storyboard testing in which users' actions on AUT are recorded and visualized in a storyboard format through application screenshots.

## *Test Complete*

Test Complete by SmartBear is a powerful commercial testing tool for web, mobile, and desktop testing. TestComplete supports various scripting languages such as JavaScript, VBScript, Python, and C++ Script. Like Katalon Studio, testers can perform keyword-driven and data-driven testing with TestComplete. The tool also offers an easy-to-use record and playback feature.

Like UFT, TestComplete's GUI object recognition capability can automatically detect and update UI objects which helps reduce the effort to maintain test scripts when the **Application under Test (AUT)** is changed. It also integrates with Jenkins in a CI process. Another interesting feature of RFT is its integration with IBM Jazz application lifecycle management systems including, IBM Rational Team Concert and Rational Quality Manager.

## *Test Plantegg Plant*

An image-based automated functional testing tool which enable the testers to interact with AUT, in the same way like end users. TestPlanteggPlant is entirely different from traditional testing tools in its approach - modeling user's point of view instead of the test scripts view often seen by testers. This allows testers with less programming skills to learn and apply test automation intuitively. The tool supports various platforms like Web, mobile, and POS systems. It offers lab management and CI integration as well.

## *Tricentis Tosca*

Tricentis Tosca is a model-based test automation tool that provides quite a broad

feature set for continuous testing including dashboards, analytics, and integrations to support agile and DevOps methodologies. Tricentis Tosca helps users to optimize the reusability of test assets. Like many other test automation tools, it supports a wide range of technologies and applications such as web, mobile, and API. Tricentis Tosca also has features for integration management, risk analysis, and distributed execution.

### *Ranorex*

Ranorex is a quite comprehensive commercial automation tool for web, mobile, and desktop testing. The tool features advanced capabilities for GUI recognition, reusable test scripts, and record/playback. Codeless test creation is a very useful feature that allows new automation testers to learn and apply test automation to the projects.

This tool supports Selenium integration for web application testing. Testers can distribute the execution of their tests across platforms and browsers using Selenium grid. Ranorex offers a low-pricing model for businesses.

### *Robot framework*

Robot Framework is an open-source automation framework which implements the keyword-driven approach for acceptance testing and **acceptance test-driven development (ATDD)**. Robot Framework provides frameworks for different test automation needs. But its test capability can be further extended by implementing additional test libraries using Python and Java. Selenium WebDriver is a popular external library used in Robot Framework.

Test engineers can leverage Robot Framework as an automation framework for not only web testing but also for Android and iOS test automation. Robot Framework can be easy to learn for testers who are familiar with keyword-driven testing.

In conclusion, the above-mentioned automation tools offer unique features in addressing the growing challenges of software automation. Most provide

capabilities for continuous testing and integration, test management, and reports.  
They all support increasing automation needs for Web and Mobile testing.

# Chapter 6

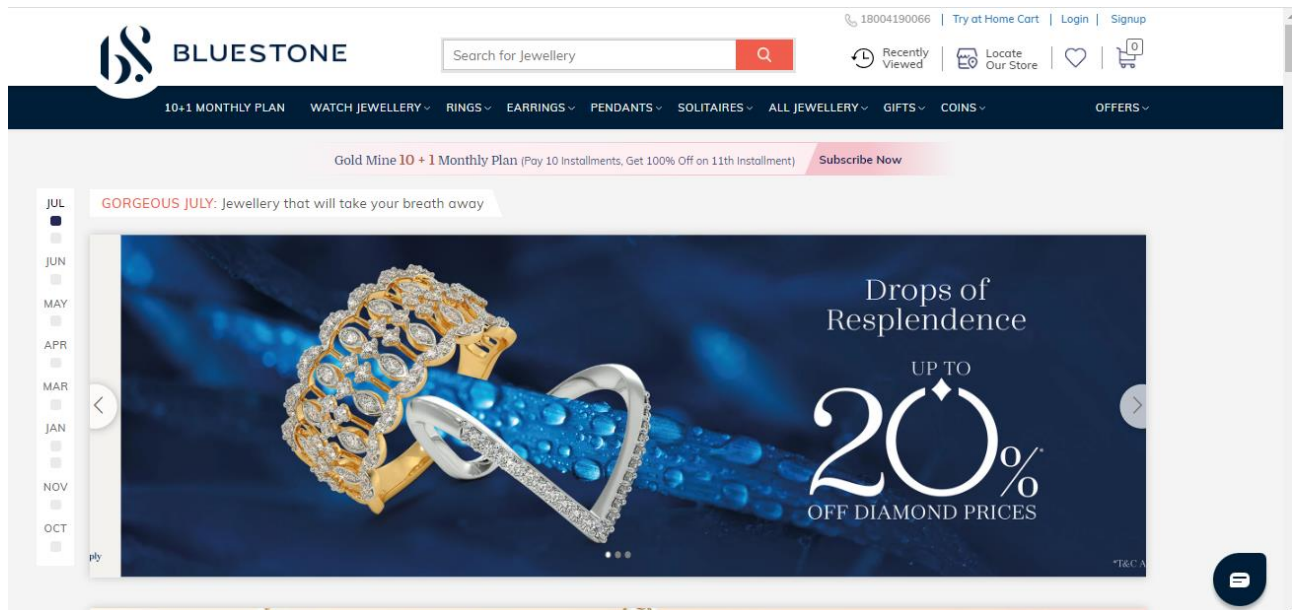
# Software Testing

## Manual Testing BLUESTONE

BlueStone is India's leading destination for high quality fine jewellery with strikingly exquisite designs. We aim at revolutionizing the fine jewellery and lifestyle segment in India with a firm focus on craftsmanship, quality and customer experience. In a short span of time, BlueStone has built a large family of loyal consumers in India and abroad.

BlueStone is a web-based e commerce application and it deals with all type of jewellery items and it has both mobile app support for customers to track orders and easy access .

## BLUESTONE HOME PAGE:



## Project Flow:

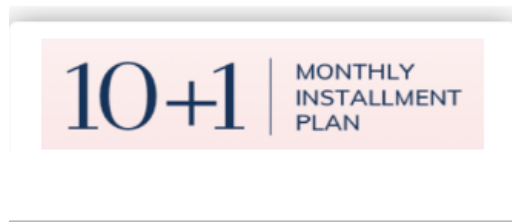
- User can search the desired product from global search
- User can register once and continue shopping and track items while delivery
- User can get updates on need basis
- User can also continue shopping using mobile app

## Product Module:

1. Gold Mine Text Verification In Terms Of Color ,Font ,Spacing Between Words, Design And Alignment



2. 10+1 , MONTHLY INSTALLMANET PLAN Text Verification In Terms Of Color ,Font ,Spacing Between Words, Design And Alignment



3. Verifying Other Text In Terms Of Color, Font, Spacing Between Words, Design And Alignment



Pay 10 installments, get 100% off  
on 11<sup>th</sup> installment!

\*Redeemable from 6th month

#### 4. Verifying Monthly Amount In Terms Of Color ,Font ,Spacing Between Words, Design And Alignment

Verify user able to enter amount in field and application should accept the user to enter amount

Enter Monthly Amount  
2000

You will pay  
only ₹ 1500

Verify user able to enter some text in amount in field and verify application accepting the user to enter data

You are eligible for additional **25% DISCOUNT** on your first installment

Amount should only contain numbers

Test

Enter Email address

START NOW

Want to pay your Gold Mine Installment? [Click to Pay](#)

#### 5. Verifying Email Address And START NOW Button Color And Functionality And Below Text In Terms Of Color ,Font ,Spacing Between Words, Design And User Able To Enter Email And Verify The Functionality Of It.

Enter Email address  
smartbridge@test.com

START NOW

Want to pay your Gold Mine Installment? [Click to Pay](#)

You are eligible for additional **25% DISCOUNT** on your first installment

Amount should only contain numbers  
Test

Please enter a valid email  
smartbridge

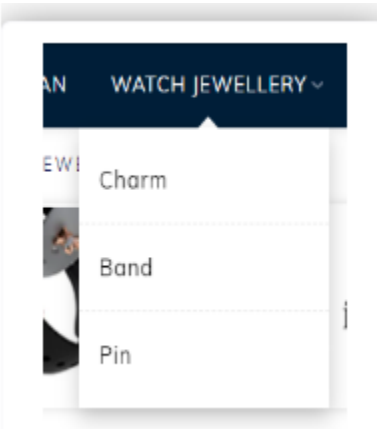
START NOW

Want to pay your Gold Mine Installment? Click to Pay

Verifying enter email address field validation when user enter invalid email


## WATCH JEWELLERY

Under Watch Jewellery for various watches with charm model , Band model, Watch PIN model



### 1. Verifying Charm Model In Terms Of Color ,Font ,Spacing Between Words, Design And Whole Representation


HOME / JEWELLERY / WATCH CHARM JEWELLERY



18kt

will be valued at

22kt



{ 22kt will be valued at 24kt }

{ 14kt will be valued at 16kt }

{ 16kt will be valued at 18kt }

Walk into any BlueStone store with your old gold and walk out with jewellery of a higher value.

\*TBC Apply

JEWELLERY | 30 Designs

PRICE

TYPE

METAL

GENDER

OFFERS

MORE FILTERS

ALL

TRY AT HOME

DESIGNS IN STORE

POPULAR

2. Verifying all the text available on screen in terms of color ,font ,spacing between words, design  
Verifying various text available on the screen to make sure all data is represented correct

22kt will be valued at 24kt

14kt will be valued at 16kt

16kt will be valued at 18kt

Walk into any BlueStone store with

your old gold and walk out with

jewellery of a higher value.

T&C Apply

3. Verifying various filter types once we apply on filter how the application is filtering out results  
and how the results are loading.

JEWELLERY | 30 Designs

PRICE

TYPE

METAL

GENDER

OFFERS

MORE FILTERS

ALL

TRY AT HOME

DESIGNS IN STORE

POPULAR

FILTERED BY

Watch Charm

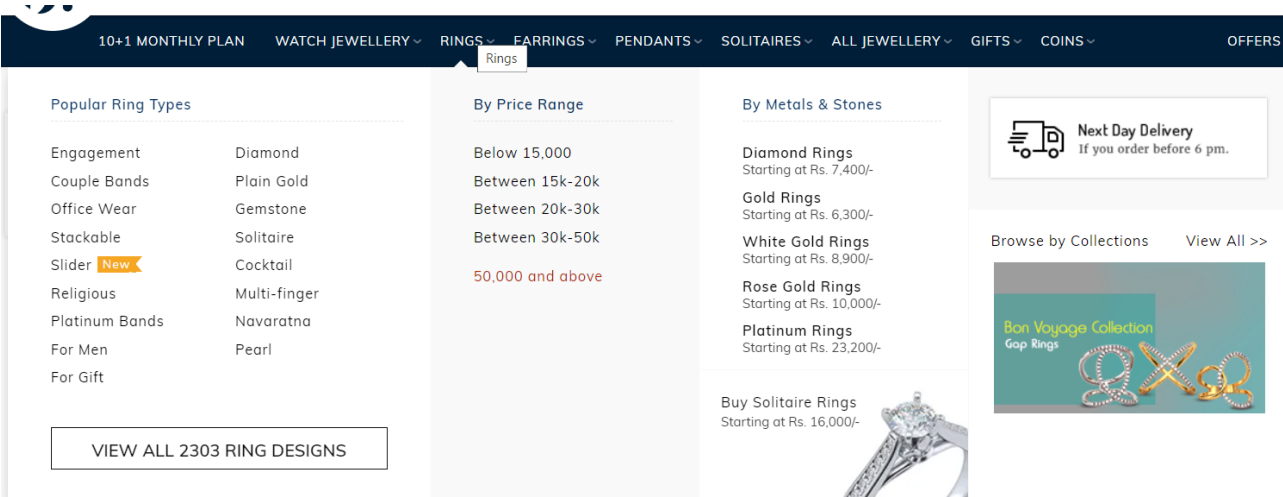
4. Verifying various images are loading properly and videos are playing when user trying to play it  
or not related to items. Verifying the different price for different Items.



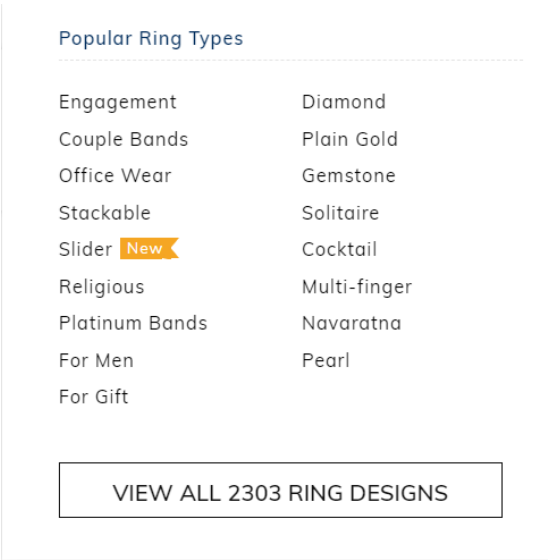
₹ 13,479

10% OFF ON MAKING CHARGE

# Verifying RINGS Module And It's Sub Modules



2. Verifying Rings Sub Modules Like Popular Ring Types And All Links Are Redirecting To Pages Or Not And All Links Are Clickable And Having Any Tags Beside To Existing Types



### 3. Verifying Link Functionalities For Each Individual Sub Modules

Verifying each and every sub module links functionality, upon clicking each link is it navigating to desired pages are not

#### Verifying links are navigating to desired page

In simple words, when user clicks on sub module links it has to navigate to desired link and show the products accordingly

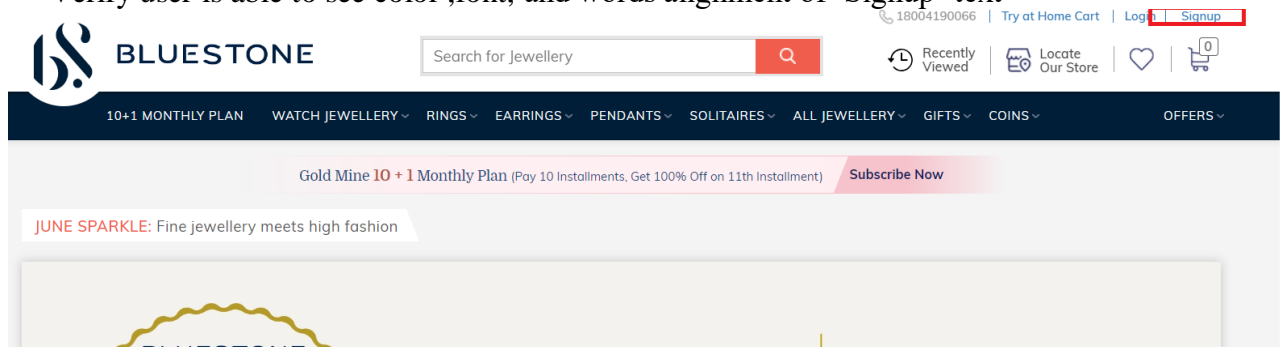
#### Verifying user able to load more results by clicking on ‘VIEW ALL <<NUMBER>> RING DESIGNS

When User clicks on ‘VIEW ALL <<NUMBER>> RING DESIGNS , it has to load all rings designs available and show images of rings with rates of that particular ring

### Verifying Signup Link Functionality

#### 1. Verifying User Is Able To Click On Signup Link And Able To See Link On Top Right Corner

Verify user is able to see color ,font, and words alignment of ‘Signup’ text



#### 2. Verifying User Able To Enter All Required Information In The Signup Form And Able To Complete Signup Process With All Positive And Negative Data

HELLO THERE, SIGNUP HERE

Name

Email

Mobile Number

☐ I accept BlueStone Terms of Service and Privacy Policy.

CREATE ACCOUNT

Already Have an Account? Login

f

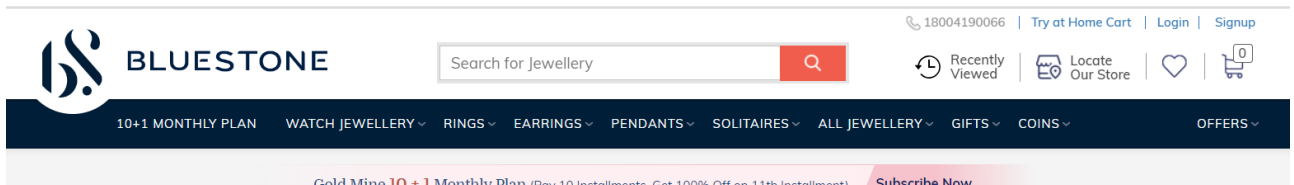
SIGNUP WITH FACEBOOK

Verifying user able to Signup with facebook account if user is having a facebook account

## Verifying Login Link Functionality

### 1. Verifying User Is Able To Click On Login Link And Able To See Link On Top Right Corner

Verify user is able to see color ,font, and words alignment of ‘Login text



2. Verifying User Able To Enter All Required Information In The Login Form And Able To Login Successfully With All Positive And Negative Data

WELCOME BACK, LOGIN HERE

Please enter your phone number or email address below. We will send you the OTP.

Email or Phone Number

SUBMIT

Do not have an account with us? [Sign Up](#)

LOG IN WITH EMAIL

f LOGIN WITH FACEBOOK

2. Verifying user able to login with email account in the Login form and able to login successfully with all positive and negative data

WELCOME BACK, LOGIN HERE

Email

Password

LOG IN

CANCEL

[Forgot Password](#)

Do not have an account with us? [Sign Up](#)

f LOGIN WITH FACEBOOK

## Final Testing Areas:

1. Test the Home page elements color, font, alignment, design
2. Test the global search
3. Test the Try Home Cart , Login, Signup link functionality
4. Test the recently viewed ,Locate store, favourites, Cart functionality
5. Test each and every module and sub module functionality is it navigating to desired page

6. Test videos related to jewellery items playing correctly or not
7. Test price for each item
8. Test filters to load results is correctly working or not
9. Test all dropdowns are correctly loading the results
10. Test loading results from dropdown are accurate or not
11. Test browser compatibility
12. Test user is able to navigate to Top section when user at bottom by clicking on TOP button
13. Test user is able to chat with executive

## **Automation Testing BLUESTONE**

Established in 2011, BlueStone is India's leading destination for high quality fine jewellery with strikingly exquisite designs. We aim at revolutionizing the fine jewellery and lifestyle segment in India with a firm focus on craftsmanship, quality and customer experience. In a short span of time, BlueStone has built a large family of loyal consumers in India and abroad.

We house more than 8000 unique designs for you to choose from. All these designs are crafted to perfection with utmost care giving you the flexibility to customize the product's gold purity and colour or diamond clarity to suit your needs.

Our stores have been instrumental in spreading the shine of BlueStone and bringing us closer to you. With world class experience, friendly staff and the dazzling beauty of exquisite jewellery, every store is a sparkling gem.

With an award-winning design team that pays great attention to detail, each of our products are a symbol of perfection. With cutting edge innovation and latest technology, we make sure the brilliance is well reflected in all our jewellery.

We also offer a 30 Day Money Back guarantee, Certified Jewellery and Lifetime Exchange. You can also experience luxury shopping from the comfort of your home with our complimentary Try At Home service.



BlueStone is a web-based e commerce application and it deals with all type of jewellery items and it has both mobile app support for customers to track orders and easy access.

## **Project Overview**

The BLUESTONE project aims to automate various tasks on the Bluestone website. It leverages Selenium, a powerful web automation tool, to interact with web elements and perform actions such as clicking buttons, filling forms, and extracting data.

## **Getting Started**

### **Prerequisites**

Before getting started with the BLUESTONE project, ensure you have the following prerequisites:

List the software and tools that need to be installed before running the project, such as

- **Python (3.x)**
- **Selenium (3.x)**
- **Pytest (Latest Version)**
- **Any other dependencies**

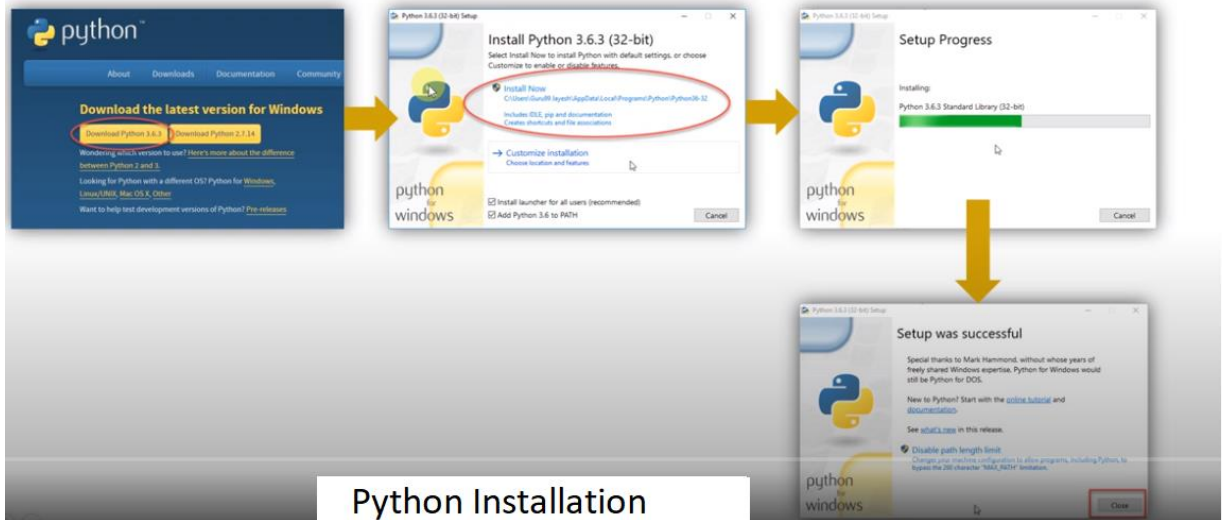
### **Installation**

To install the necessary dependencies for the BLUESTONE automation project, follow these steps:

**Install Python:** Visit the Python website (<https://www.python.org>) and download the latest version of Python. Follow the installation instructions for your operating system.

# Download and install python on Windows

■ <https://www.python.org/downloads/>



Download and install gitbash from <https://git-scm.com/download/win>

git-scm.com/download/win

**git** --local-branching-on-the-cheap

Search entire site...

**About**  
**Documentation**  
**Downloads**  
GUI Clients  
Logos  
**Community**

The entire **Pro Git** book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

## Download for Windows

Click here to download the latest (2.41.0) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 1 day ago, on 2023-07-07.

### Other Git for Windows downloads

- Standalone Installer
- 32-bit Git for Windows Setup.
- 64-bit Git for Windows Setup. ←
- Portable ("thumbdrive edition")
- 32-bit Git for Windows Portable.
- 64-bit Git for Windows Portable.

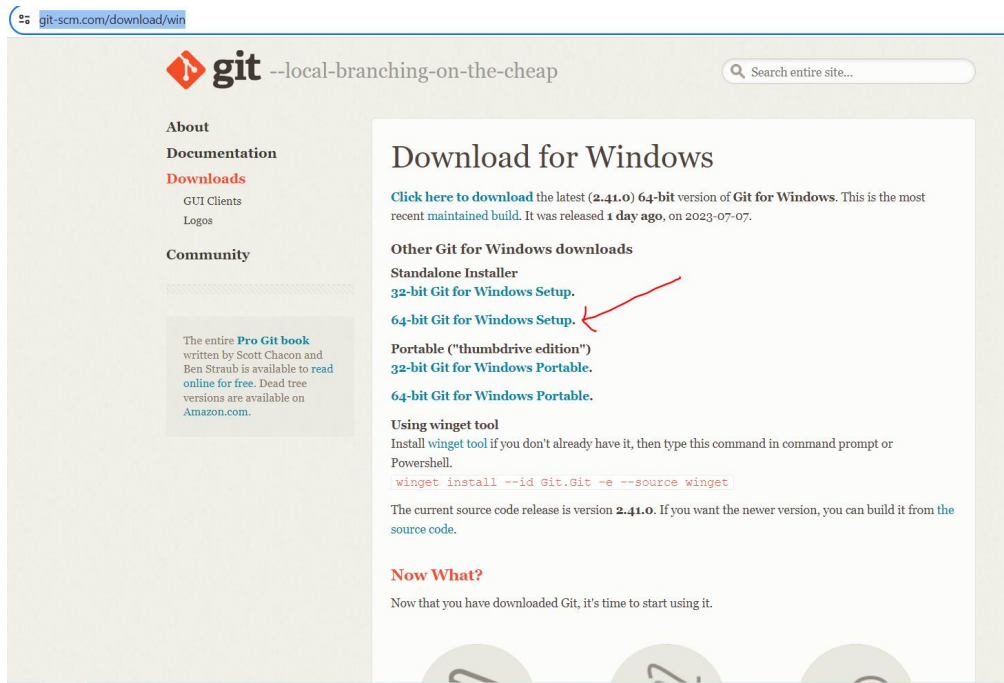
**Using winget tool**  
Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version 2.41.0. If you want the newer version, you can build it from the source code.

### Now What?

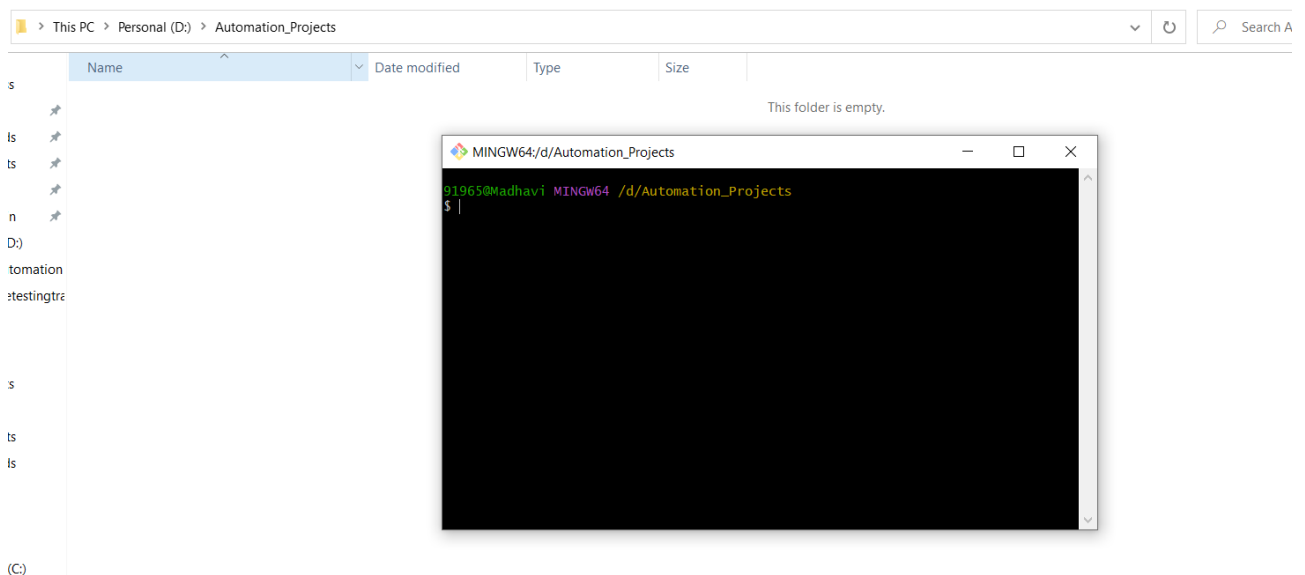
Now that you have downloaded Git, it's time to start using it.



Go to the desired location where you want to create the folder, such as Desktop.

Folder name like eg...(Automation\_Testing\_Project)

Open the folder-???? Right click with in the folder and select open gitbash here, you will get pop up like below and clone the repository based on the next steps provided below.



## Project Setup

Explained how to set up the project locally, including steps like:

**Activity 3.1:-** Clone the repository from github.


**Activity 3.2:-** Clone the BLUESTONE project repository from [repository URL].

[https://github.com/viswatesting123/SmartBridge\\_Automation\\_Projects.git](https://github.com/viswatesting123/SmartBridge_Automation_Projects.git)

**Activity 3.3:-** Install all dependencies using requirements.txt

Requirements.txt containing all software's/libraries required for our projects. We don't need to install separately.

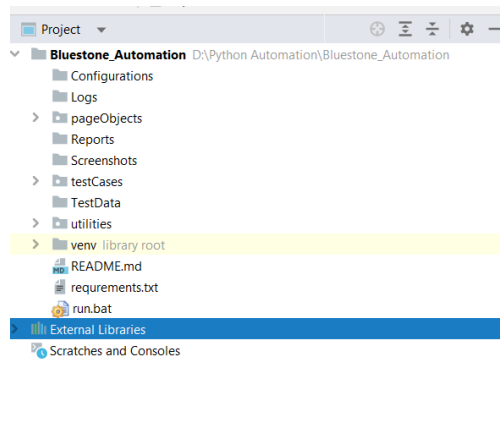
**Activity 3.4:-** RUN the command from the terminal:

 Copy code

```
pip install -r requirements.txt
```

## Project Structure

Explained The Structure Of The Project, Including Directories And Files, And Their Purposes. For Example:

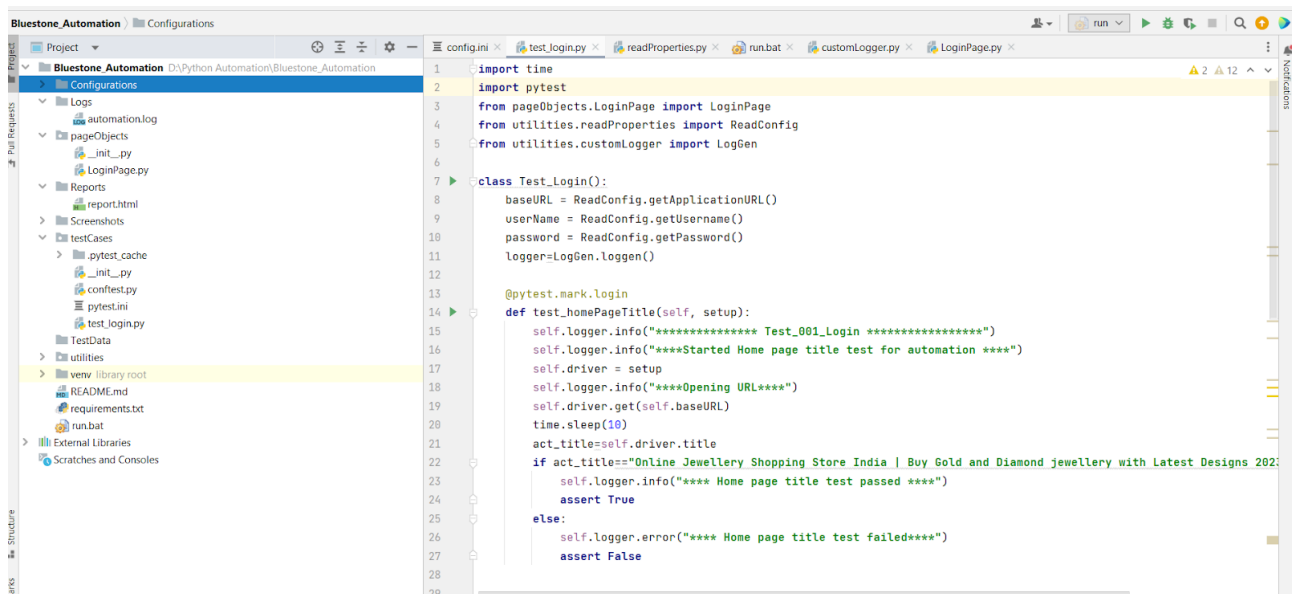


- testcases/: Directory containing test scripts
- pageObjects/: Directory containing page object models
- utils/: Directory containing utility functions
- reports/: Directory containing test reports
- testData/: Directory containing test data (ex: excel, csv, text, json files) related to project
- Configuration/: Configuration file for storing settings
- Screenshots/: Project testcases screenshots.
- Logs/: Directory maintain the project logs.
- Run.bat/: Now, you can double-click the run.bat file to execute the Python script specified in the command. The script will run in a Command Prompt window.

## Writing Test Cases

Explained how to write test cases using Python, Selenium, and pytest. Provide examples of different types of test cases, such as:

- Test case 1: Home Page functionality



The screenshot shows a code editor with a project structure on the left and a Python script in the main editor. The project structure includes directories like Configurations, Logs, automation.log, pageObjects, Reports, Screenshots, testCases, testData, utilities, and a venv directory. The Python script, located in test\_login.py, defines a Test\_Login class with a setUp method and a test\_homePageTitle method. The test method uses Selenium to open a browser, navigate to a URL, and verify the page title.

```

1 import time
2 import pytest
3 from pageObjects.LoginPage import LoginPage
4 from utilities.readProperties import ReadConfig
5 from utilities.customLogger import LogGen
6
7 class Test_Login():
8     baseURL = ReadConfig.getApplicationURL()
9     userName = ReadConfig.getUsername()
10    password = ReadConfig.getPassword()
11    logger=LogGen.loggen()
12
13    @pytest.mark.login
14    def test_homePageTitle(self, setup):
15        self.logger.info("***** Test_001_Login *****")
16        self.logger.info("****Started Home page title test for automation ****")
17        self.driver = setup
18        self.logger.info("****Opening URL****")
19        self.driver.get(self.baseURL)
20        time.sleep(10)
21        act_title=self.driver.title
22        if act_title=="Online Jewellery Shopping Store India | Buy Gold and Diamond jewellery with Latest Designs 202":
23            self.logger.info("**** Home page title test passed ****")
24            assert True
25        else:
26            self.logger.error("**** Home page title test failed****")
27            assert False
28
29

```

- Test case 2: Register new user functionality
- Add new testcases as per Manual testcases prepared.

Include best practices for writing maintainable and readable test cases.

## Running Tests

Explained how to run the test cases using pytest. Include information about command-line options and flags that can be used to customize test execution.

### Run.Bat/:

Now, you can double-click the run.bat file to execute the Python script specified in the command. The script will run in a Command Prompt window.

## Test Reporting

Explained how test reports are generated and where they are stored. Include information about any tools or libraries used for reporting, such as `pytest-html`.

### Reports/:

Directory containing test reports

</

## **Troubleshooting**

List common issues that may occur during project setup or test execution and provide possible solutions.

# Chapter 7



## **Conclusion**

The BLUESTONE automation project provides a framework for automating various tasks within the BLUESTONE web application using Python, Selenium, and Pytest. With this documentation, you should be able to set up the project, execute tests, and extend/customize it according to your needs.

## References

- [1] Ruparelia, N. B. (2010). Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes*, 35(3), 8-13.
- [2] Alsmadi, I. (Ed.). (2012). *Advanced Automated Software Testing: Frameworks for Refined Practice: Frameworks for Refined Practice*. IGI Global.
- [3] Amaricai, S., & Constantinescu, R. (2014). Designing a Software Test Automation Framework. *Informatica Economica*, 18(1).
- [4] Chang, E., & Dillon, T. S. (1997). Automated usability testing. In *Human-Computer Interaction INTERACT'97* (pp. 77-84). Springer, Boston, MA.
- [5] Dustin, E., Rashka, J., & Paul, J. (1999). *Automated software testing: introduction, management, and performance*. Addison-Wesley Professional.
- [6] Farid, M. R., & Abraham, K. (2010). *Automated Database Applications Testing: Specification Representation for Automated Reasoning* (Vol. 76). World Scientific.
- [7] Galin, D. (2004). *Software quality assurance: from theory to implementation*. Pearson Education India.
- [8] Kasurinen, J., Taipale, O., & Smolander, K. (2010). Software test automation in practice: empirical observations. *Advances in Software Engineering*, 2010.
- [9] Kit, E. (1995). *Software testing in the real world: improving the process*. Addison-wesley.
- [10] Laukkanen, P. (2006). Data-driven and keyword-driven test automation frameworks. *M. Tech thesis, Department of Computer Science and Engineering Software Business and Engineering Institute, Helsinki University of Technology*.
- [11] Lewis, W. E. (2017). *Software testing and continuous quality improvement*. Auerbach publications.
- [12] Ramamoorthy, C. V., & Ho, S. B. F. (1975, April). Testing large software with automated software evaluation systems. In *ACM SIGPLAN Notices* (Vol. 10, No. 6, pp. 382-394). ACM.
- [13] Ramler, R., & Wolfmaier, K. (2006, May). Economic perspectives in test automation: balancing automated and manual testing with opportunity cost. In *Proceedings of the 2006 international workshop on Automation of software test* (pp. 85-91). ACM.
- [14] Sei, L. M. (2015). Automating Test Activities: Test Cases Creation, Test Execution, and Test Reporting with Multiple Test Automation Tools. *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 9(10), 2213-2216.
- [15] Testrail (2018). "Testrail." from <http://www.gurock.com/testrail/>.
- [16] Kerzazi, N., & Khomh, F. (2014, September). Factors impacting rapid releases: an industrial case study. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement* (p. 61). ACM.
- [17] Wiklund, K., Eldh, S., Sundmark, D., & Lundqvist, K. (2017). Impediments for software test automation: A systematic literature review. *Software Testing, Verification and Reliability*, 27(8), e1639.

- [18] Fewster, M., & Graham, D. (1999). *Software test automation: effective use of test execution tools*. ACM Press/Addison-Wesley Publishing Co.
- [19] Dustin, E., Rashka, J., & Paul, J. (1999). *Automated software testing: introduction, management, and performance*. Addison-Wesley Professional.
- [20] Goucher, A., & Riley, T. (2009). *Beautiful Testing: Leading Professionals Reveal How They Improve Software*. " O'Reilly Media, Inc."
- [21] Jorgensen, P. C. (2013). *Software testing: a craftsman's approach*. Auerbach Publications.
- [22] Garg, N. (2014). *Test Automation using Selenium WebDriver with Java: Step by Step Guide*. Test Automation Using Selenium with Java.

- [23] Avasarala, S. (2014). *Selenium WebDriver practical guide*. Packt Publishing Ltd.
- [24] Collin, M. (2015). *Mastering selenium webdriver*. Packt Publishing Ltd.
- [25] Mg, R. P. (2015). *Learning Selenium Testing Tools*. Packt Publishing Ltd.
- [26] Gojare, S., Joshi, R., &Gaigaware, D. (2015). Analysis and design of selenium webdriver automation testing framework. *Procedia Computer Science*, 50, 341- 346.