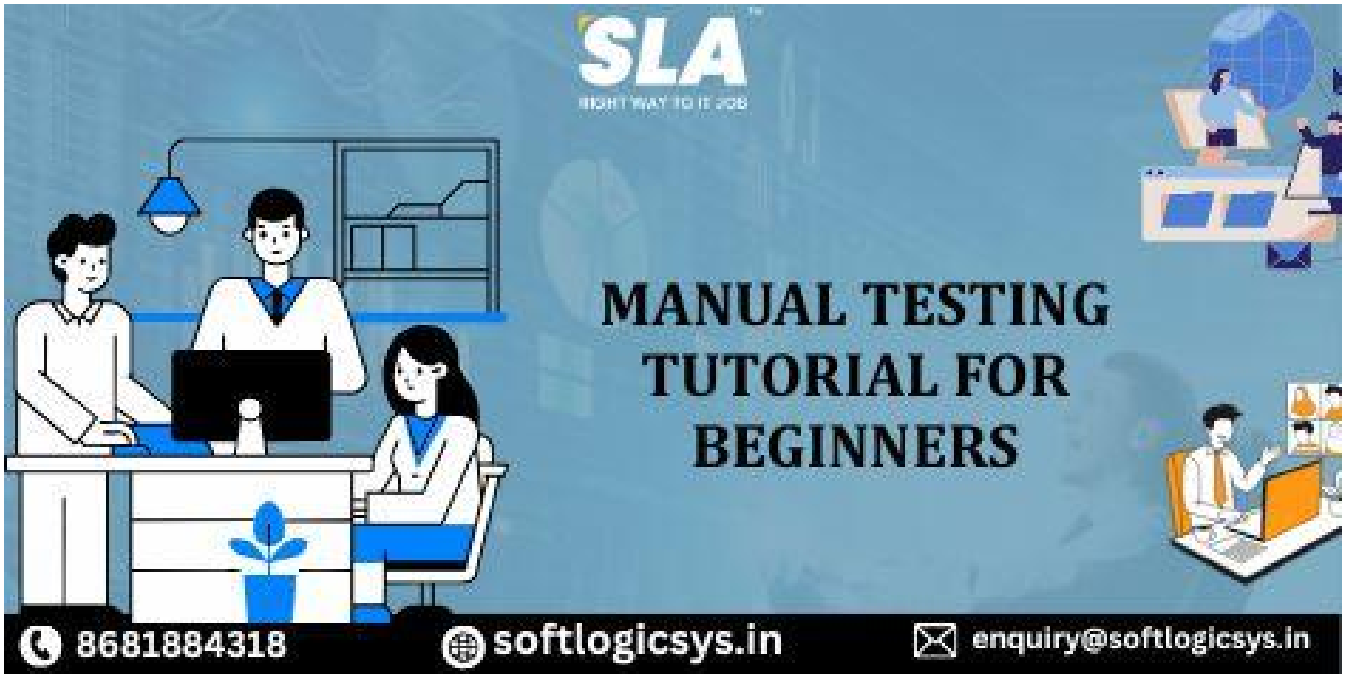

Manual Testing Tutorial for Beginners

Introduction

Are you confused about how software is verified, or are you puzzled about where to begin your career in QA? To most novices, all the documentation and techniques can be overwhelming.

This manual testing tutorial will get you started in the field of quality assurance. We teach you all the fundamental techniques involved in writing test cases and executing them to build high-quality software with confidence. Click here to see the **Manual Testing Course Syllabus**!

Why Students or Freshers Learn Manual Testing?

Below are the reasons to learn Manual Testing as a fresher:

- **Low Barrier to Entry:** Manual Testing requires no coding experience initially; thus, it's the fastest route for beginners to enter the IT industry and start contributing immediately.
- **Core QA Foundation:** All successful automation and performance testing careers are built on a solid understanding of manual testing principles, including test case design and bug reporting.
- **Understanding User Experience:** Manual testers develop a critical eye for usability, thinking like an end-user to make sure the software meets real-world needs and specifications.
- **Great for Critical Thinking:** It forces you to explore, break, and validate software in depth, thus honing your analytical and logical thinking.

Ready to land your first QA position? Click here for **Manual Testing Interview Questions and Answers**!

Step-by-Step Manual Testing Tutorial for Beginners

Manual Testing is the process of testing the software for defects without using any automated tools. It is, in fact, the basis of all QA. In this tutorial, one will be taken through the complete life cycle of manual testing.

Step 1. Prerequisites and Setup

Since manual testing relies on human observation and interaction, there are no specific installations required for the testing itself. However, you will need a few essential tools for documentation and bug tracking.

1.1 Setup Essential Tools

- **Browser:** Install different modern browsers like Chrome, Firefox, and Edge for cross-browser testing.
- **Text Editor:** Any text editor, such as Notepad++, VS Code, or Google Docs, will do for writing test cases.
- **Bug Tracking Tool:** Create a free account or familiarize yourself with tools such as Jira, Trello, or Bugzilla for logging defects.
- **Requirement Document:** Get a copy of the SRS or User Stories for the application under test. This document identifies what the software is supposed to perform.

Step 2. STLC – Understanding the Software Testing Life Cycle

Manual testing is a structured process based on STLC, which is often merged with the Software Development Life Cycle.

2.1 Requirement Analysis

This is the most important step. The testers go through the requirement documents, which are SRS and user stories that describe the system functionalities, the non-functional features (which are generally performance, security), and user expectations.

- **Objective:** Highlight testable requirements and ensure ambiguous requirements are explained to stakeholders (Business Analyst/Developer).
- **Deliverable:** List of questions/clarifications, and the development of RTM (optional for small projects) in order to map requirements to test cases.

2.2 Test Planning

Based on the analysis, the QA Lead or Senior Tester defines the strategy and scope.

- **Scope:** What to test and, critically, what not to test.

- **Strategy:** Defines the test types, such as functional, regression, security, etc., and the resources required-manpower and environment.
- **Deliverable:** Test Plan Document.

Step 3: Designing Test Cases

A test case is a set of conditions or actions under which a tester determines whether a system under test satisfies a requirement. This is the core skill of a manual tester.

3.1 Test Case Structure

Every good test case should include the following elements:

Element	Description
Test Case ID	Unique identifier (e.g., TC_LOGIN_001).
Test Case Title	A brief, descriptive title (e.g., Verify Valid Login).
Pre-Conditions	Conditions that must be met before executing the test (e.g., User must be registered).
Steps	Detailed, sequential actions the tester performs.
Expected Result	The precise outcome the system should produce.
Post-Conditions	State of the system after test execution.

3.2 Sample Test Case: Login functionality

Let's design a test case for successful login:

Test Case ID: TC_LOGIN_001 Test Case Title: Verify user can successfully log in with valid credentials. Pre-Conditions: User "testuser@example.com" exists and has password "Pass@123". Steps:

1. Navigate to the application's login page (URL: www.app.com/login).

2. Enter valid username: testuser@example.com

3. Enter valid password: Pass@123

4. Click the "Login" button.

Expected Result:

1. The user is redirected to the dashboard page (URL: www.app.com/dashboard).

2. The welcome message "Welcome, testuser!" is displayed.

3.3 Test Case Design Techniques

Techniques to be used to ensure that coverage is comprehensive:

- **Equivalence Partitioning:** Segment input data into partitions. Since all other values within a partition are presumed to produce the same outcome, only one value from each partition is tested.
    - **Example:** For the 18 to 60 age input field, test one value in the range, for instance 30, one below the range, say 17, and one above the range, say 61.
- **Boundary Value Analysis (BVA):** This testing should be performed at the boundaries of the valid partitions, since a lot of the defects come at these extremities.
    - **Example:** For the 18-60 age field, test 17, 18, 59, 60, and 61.

- **Error Guessing:** Utilize intuition and experience to identify where the developer might have made mistakes, such as: division by zero, empty fields (fields that should not be empty), and special characters in number fields.

Step 4. Setup of Test Environment

The testing environment should mimic production to ensure the tests are accurate.

- **Checklist:** Check OS, browsers and its versions, database connectivity, test data required.
- **Data Preparation:** Create/maintain test data necessary for execution, like a list of valid/invalid customer IDs and specific product configurations.

Step 5. Test Execution and Reporting

This is the phase when you will execute the test cases against the Build, which is the testable software version provided by the developers.

5.1 Test Case Execution

1. Follow the Pre-Conditions and Steps exactly as described in the test case.
2. Compare the Actual Result with the Expected Result.
3. **Mark the Status:**
    - **PASS:** Actual Result matches Expected Result.
    - **FAIL:** Actual Result does not match Expected Result-this is indicative of a defect/bug.
    - **BLOCKED:** The implementation of the test cannot be run due to a dependency with a failed or unresolved issue.
    - **SKIP:** The test is out of scope for the current build/release.

5.2 Defect Logging (Bug Reporting)

When a test case fails, it is necessary to log a defect in the bug tracking tool, such as Jira, right away. A well-written bug report is crucial in assisting developers in finding the solution quickly.

Key components of a defect report:

- **Defect ID:** Unique identifier generated by the tool.
- **Summary:** A short, clear title such as “As a user I am unable to log in using a valid username”.
- **Steps to Reproduce:** Simple, clear, numbered steps that guarantee that a developer can reproduce the bug.
- **Expected Result:** What should have happened.
- **Actual Result:** What did happen – the error.
- **Severity:** The impact of the defect on the system. Examples are Critical, Major, and Minor.
- **Priority:** The priority of how soon the defect should be fixed. P1-P4
- **Environment:** Browser/OS/URL where the defect was found.
- **Attachments:** Screenshots or video recordings of the failure.

5.3 Defect Life Cycle

Defects go through a life cycle before closure: New > Assigned > Open > Fixed > Retest > Closed/Reopen. Being a manual tester, your main role would be to log the bug and do the Retest when the developer marks it as Fixed.

Step 6. Regression and Exit Criteria

6.1 Regression Testing

While fixing any bug or adding any feature, a developer might unconsciously break some of the already working functionality. Regression Testing is re-running a selected set of previously passed test cases to ensure that recent changes have not introduced new defects. This is a continuous process throughout the development cycle.

6.2 Test Closure

The testing cycle is complete when the Exit Criteria are satisfied:

- All planned tests are executed.
- A fixed percentage of critical/major defects are closed.
- The client/stakeholder passes the acceptance tests.
- The general status of quality is acceptable.

Step 7. Types of Manual Testing

Working as a manual tester, you will conduct several types of testing:

- **Functional Testing:** The testing of the core features against requirements, such as whether the payment button works.
- **Usability Testing:** This involves checking how easy the application is for the end-user; that is, whether navigation is intuitive and the layout clean.
- **Exploratory Testing:** The process of testing without formal test cases. Instead, using your intuition to explore the application and find hidden defects.
- **System Testing:** Testing the entire, integrated system to verify that it meets all specified requirements.
- **Acceptance Testing UAT:** Formal testing intended to validate that the system meets the customer's needs and is ready for deployment.

Now, you have covered the complete process of Manual Testing: from initial analysis and test case design to execution and bug reporting. A strong foundation toward a successful QA career requires this.

Ready to put these steps into practice and master real-world testing scenarios? Click here for **Manual Testing Challenges and Solutions**!

Real Time Examples for Manual Testing Tutorial for Learners

Following are some real-time examples showing the key significance of Manual Testing in the software development lifecycle:

Usability Testing on a Mobile Banking Application

Goal: Make the app intuitive, accessible, and easy to use by all customers in performing critical actions such as transferring money or paying bills.

Manual Testing Approach: Exploratory Testing will be performed by testers on different types of devices (iOS, Android, and other screen sizes) and network conditions (Wi-Fi, 4G).

- He will look for usability heuristics-whether the size of buttons is big enough to press by fingers, whether error messages will be clear, and navigation requires minimum clicks.
- Therefore, techniques such as Negative Testing-checking whether the amount to be transferred is greater than the balance-can be performed. It requires a human touch to judge UX.

GUI Testing and System Integration for CRM Tool:

Goal: To ensure the complex GUI components-for example, forms, dashboards, reports-render correctly on different browsers and that newer features dovetail seamlessly with already deployed backend systems.

Manual Testing Approach: The testers would run detailed Test Cases for each and every visible element to ensure that colors, fonts, alignment, and responsiveness are correct.

- **Cross-Browser Testing:** They do System Integration Testing by manually filling out a complicated form and immediately verifying the associated reports and database records to make sure that data flow from front-end to back-end services is correct.

Ad-hoc and Destructive Testing on a Gaming Server Patch:

Goal: Uncover unforeseen bugs and security loopholes after a major patch or update before release to millions of players.

Manual Testing Approach: The testers do Ad-hoc Testing, which means unstructured, spontaneous testing, trying to do strange and illogical things, like clicking “Save” many times in a row or disconnecting the network during saving, trying to use invalid characters for usernames.

This kind of testing usually catches critical, unexpected bugs that automated scripts fail to notice because it depends on the tester’s critical and lateral thinking to intentionally attempt to “break the system.”

Ready to put theory into practice and build a solid portfolio in QA? Click here for **Manual Testing Project Ideas**!

FAQs About Manual Testing Tutorial for Beginners

1. What are the 7 phases of STLC?

The main phases of the STLC are Requirement Analysis, Test Planning, Test Case Development, Test Environment Setup, Test Execution, Test Cycle Closure, and sometimes Test Maintenance. These provide the well-structured steps for quality assurance.

2. What are QA manual testing?

QA Manual Testing refers to the process of testing software for quality by executing test cases manually without depending on any automation testing tool. In this process, the tester plays the role of an end-user to find bugs, validate functionality, ensure requirements compliance, and judge the overall UX.

3. Is Manual Testing easy to learn?

Yes, Manual Testing is relatively easy to learn since it requires minimum technical skills or coding to get started. Here, the focus is on critical thinking, analytical skills, and

understanding the requirements, which makes it a very good starting point for beginners in the IT industry.

4. What is SDLC vs STLC?

QA Manual Testing refers to the process of testing software for quality by executing test cases manually without depending on any automation testing tool. In this process, the tester plays the role of an end-user to find bugs, validate functionality, ensure requirements compliance, and judge the overall UX.

5. Is Manual Testing easy to learn?

Yes, Manual Testing is relatively easy to learn since it requires minimum technical skills or coding to get started. Here, the focus is on critical thinking, analytical skills, and understanding the requirements, which makes it a very good starting point for beginners in the IT industry.

6. What is SDLC vs STLC?

SDLC basically denotes the overall process of managing the software creation right from its inception to deployment. STLC, on the other hand, is a subset of SDLC and defines sequential steps and activities related to testing and quality assurance.

7. Is QA an entry level job?

Yes, QA is considered to be a common entry-point position within the tech industry. Manual Tester and Junior QA Analyst positions require foundational skills that can be learned in a short period of time, making it a defined career path onto Automation Engineer, QA Lead, or Performance Tester.

8. What's the salary for manual testers?

Manual Testing Salary for Freshers and Experienced can vary widely depending on location and experience. Manual Testers start somewhere in the range of ₹3 to ₹5 Lakhs per annum in India, and experienced/senior manual testers make significantly more with domain expertise.

9. Will AI replace manual testers?

No, AI won't replace manual testers. AI tools will automate repetitive or data-intensive types of testing. However, the role of a manual tester in Exploratory Testing, in the validation of Usability, and in exercising critical human judgment is irreplaceable.

10. Is manual testing a good future?

Manual Testing is a good foundational skill that can provide a great future. The purely manual jobs will eventually decrease, but all the skills you learn-test design, critical analysis, bug reporting-are critical for converting and going into Automation Testing, DevOps, or QA Management, which are high-demand areas.

11. Which AI tool is best for QA testing?

There is no single "best" AI tool, as suitability depends on the project. Tools like Testim.io, Cypress Studio, or Applitools (Visual AI) use machine learning to assist with script creation, maintenance, and visual testing, significantly boosting tester efficiency.

12. What is L1, L2, and L3 testing?

L1, L2, and L3 are levels of support/defect resolution, not types of formal testing. L1 is the initial bug triage, while L2 is the in-depth analysis/replication. In L3, complex code or architectural issues are fixed by the development team.

Conclusion

You have successfully completed this beginner's guide to Manual Testing. You now possess the core knowledge of the STLC, test case design, and effective bug reporting. Remember that a QA professional draws much of their strength from critical thinking and commitment to quality. Continue to practice these skills in order to develop an eye for defects. Foundational knowledge like this is crucial to move on to automation and specialized testing roles. Ready to solidify these skills and gain practical hands-on experience? Enroll in our comprehensive **Manual Testing Course in Chennai** and launch your career in QA!