**Test cases:**

**Step 1:** Test Case ID

Test cases should all bear unique IDs to represent them. In most cases, following a convention for this naming ID helps with organization, clarity, and understanding.

**Step 2:** Test Description

This description should detail what unit, feature, or function is being tested or what is being verified.

**Step 3:** Assumptions and Pre-Conditions

This entails any conditions to be met before test case execution. One example would be requiring a valid Outlook account for a login.

**Step 4:** Test Data

This relates to the variables and their values in the test case. In the example of an email login, it would be the username and password for the account.

**Step 5:** Steps to be Executed

These should be easily repeatable steps as executed from the end user’s perspective. For instance, a test case for logging into an email server might include these steps:

1. Open email server web page.
2. Enter username.
3. Enter password.
4. Click “Enter” or “Login” button.

**Step 6:** Expected Result

This indicates the result expected after the test case step execution. Upon entering the right login information, the expected result would be a successful login.

**Step 7:** Actual Result and Post-Conditions

As compared to the expected result, we can determine the status of the test case. In the case of the email login, the user would either be successfully logged in or not. The post-condition is what happens as a result of the step execution such as being redirected to the email inbox.

**Step 8:** Pass/Fail

Determining the pass/fail status depends on how the expected result and the actual result compare to each other.

Same result = Pass  
Different results = Fail

Story points should reflect the **total** amount of time that a work item will take — including any design, implementation, and testing efforts related to creating potentially-shippable product every iteration.

Black Box Testing is **also known as functional testing, data-driven testing, and closed box testing**. White Box Testing is also known as structural testing, clear box testing, code-based testing, and transparent testing. Grey Box Testing is also known as translucent testing as the tester has limited knowledge of coding.

What is meant by defect life cycle?

Bug life cycle also known as defect life cycle is a process in which **defect goes through different stages in its entire life**. This lifecycle starts as soon as a bug is reported by the tester and ends when a tester ensures that the issue is fixed and won't occur again.

<https://www.guru99.com/defect-life-cycle.html>

**what is test plan**

A Test Plan refers to **a detailed document that catalogs the test strategy, objectives, schedule, estimations, deadlines, and the resources required for completing that particular project**. Think of it as a blueprint for running the tests needed to ensure the software is working properly – controlled by test managers.

**Who will write??**

Writing a test plan is typically a test management or leadership responsibility. Others on the test team and in the organization (such as users and developers) may have input and review tasks, but it is generally **up to the manager** to actually write the test plan.

**What is test plan ?**  
browserstack.com/guide/test-planning

**Test Matrix…?**

A test matrix is **used to capture the actual quality, the effort, the plan, resources and time required to complete all phases of software testing**. A coverage matrix, also known as a traceability matrix, maps the test cases and customer requirements.

**Test Artifacts:**

Test Artifacts are simply integral part of software testing. These are generally set of documents, which software project tester gets during STLC (Software Testing Life Cycle). Test artifacts are **by-products that are generated or created while performing software testing**

What is meant by UAT testing?

**User Acceptance Testing** (UAT) is the final stage of any software development life cycle. This is when actual users test the software to see if it is able to carry out the required tasks it was designed to address in real-world situations. UAT tests adherence to customers' requirements.

**Critical issues faced::**

* [#1 – Last-Minute Changes to Requirements](https://www.browserstack.com/guide/challenges-faced-by-qa#toc0)
  + [Solution](https://www.browserstack.com/guide/challenges-faced-by-qa#toc1)
* [#2 – Inadequate information on user stories](https://www.browserstack.com/guide/challenges-faced-by-qa#toc2)
  + [Solution](https://www.browserstack.com/guide/challenges-faced-by-qa#toc3)
* [#4 – Inadequate collaboration between testers and developers](https://www.browserstack.com/guide/challenges-faced-by-qa#toc5)
  + [Solution](https://www.browserstack.com/guide/challenges-faced-by-qa#toc6)
  + ===============================

**What is RCA??**

Root cause analysis (RCA) is **the process of discovering the root causes of problems in order to identify appropriate solutions**. RCA assumes that it is much more effective to systematically prevent and solve for underlying issues rather than just treating ad hoc symptoms and putting out fires.

Agile process:

It contains six phases: **concept, inception, iteration, release, maintenance, and retirement**. The Agile life cycle will vary slightly depending on the project management methodology chosen by a team. For example, Scrum teams work in short time periods known as sprints, which are similar to iterations.

Burndown projects:

A burn down chart is a graphical representation of work left to do versus time. The outstanding work is often on the vertical axis, with time along the horizontal. Burn down charts are a run chart of outstanding work. It is useful for predicting when all of the work will be completed.

How to measure Quality of the project??

[[](https://www.google.com/search?q=How+do+you+measure+quality+in+software+testing?&rlz=1C1GCEU_enIN918IN918&sxsrf=AOaemvKxaug5WPm9q0oAa7T6iGQy3sZtZg:1632809192593&tbm=isch&source=iu&ictx=1&fir=Mc-tG0VanbRFOM%252CTbN4OWwW23qD5M%252C_&vet=1&usg=AI4_-kT7jdLaERxR-M_ztsCukE6CkiF5jA&sa=X&ved=2ahUKEwjh0qOfgKHzAhWGup4KHW45AcoQ9QF6BAgKEAE#imgrc=Mc-tG0VanbRFOM)](https://www.google.com/search?q=How+do+you+measure+quality+in+software+testing?&rlz=1C1GCEU_enIN918IN918&sxsrf=AOaemvKxaug5WPm9q0oAa7T6iGQy3sZtZg:1632809192593&tbm=isch&source=iu&ictx=1&fir=Mc-tG0VanbRFOM%252CTbN4OWwW23qD5M%252C_&vet=1&usg=AI4_-kT7jdLaERxR-M_ztsCukE6CkiF5jA&sa=X&ved=2ahUKEwjh0qOfgKHzAhWGup4KHW45AcoQ9QF6BAgKEAE" \l "imgrc=Mc-tG0VanbRFOM)

**Test Effectiveness**

1. Metrics Based: Test Effectiveness Using Defect Containment efficiency.
2. Context Based: Test Effectiveness Using Team Assessment.
3. Test Cases by Requirement.
4. Defects per Requirement (Requirement Defect Density)
5. Actual Cost of Testing.
6. Budget Variance.
7. Schedule Variance.
8. Cost Per Bug Fix.