**/’------Manual Testing Concept-------**

**Manual Concept for ETL Testing**

* Software Development Life Cycle – (SDLC)
* What is Requirement Traceability Matrix (RTM)
* Software Testing Life Cycle – (STLC)
* Sanity and Smoke Testing.
* Regression Test.
* Test Strategy
* Test Plan Design
* Test Cases Design
* Defect Tracking and Reporting
* Bug/Defect Life Cycle.
* Severity and priority
* Test Closure Report
* Test summary report

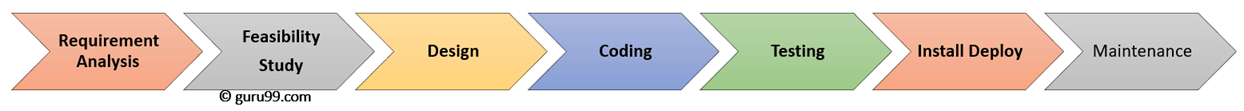
1. **Software Development Life Cycle – (SDLC)**

**SDLC** is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software that meets customer expectations.

Every phase of the SDLC life Cycle has its own process and deliverables that feed into the next phase. SDLC stands for **Software Development Life Cycle** and is also referred to as the Application Development life-cycle.

**SDLC Phases**

The entire SDLC process divided into the following SDLC steps:



SDLC Phases

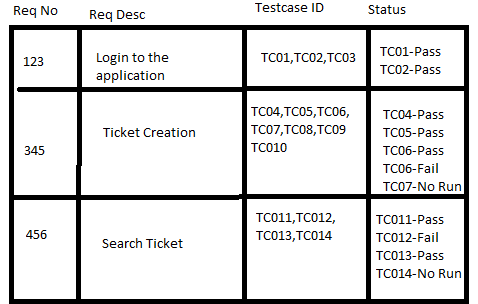
* Phase 1: Requirement collection and analysis
* Phase 2: Feasibility study
* Phase 3: Design
* Phase 4: Coding
* Phase 5: Testing
* Phase 6: Installation/Deployment
* Phase 7: Maintenance

**2.) What is Requirement Traceability Matrix (RTM)**

* **Requirement Traceability Matrix (RTM)** is a document that maps and traces user requirement with test cases.
* It captures all requirements proposed by the client and requirement traceability in a single document, delivered at the conclusion of the Software development life cycle.
* The main purpose of Requirement Traceability Matrix is to validate that all requirements are checked via test cases such that no functionality is unchecked during Software testing.
* Traceability Matrix is nothing but a mapping between the requirement and Test cases. How many test cases written for particular Requirement and all the test cases for specified requirement?

**Which Parameters to include in Requirement Traceability Matrix?**

* Requirement ID
* Requirement Type and Description
* Test Cases with Status



Above is a sample requirement traceability matrix.

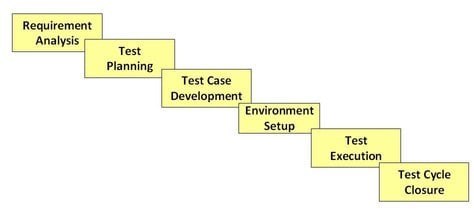
**3.Software Testing Life Cycle – (STLC)**

* STLC stands for Software Testing Life Cycle. STLC is a sequence of different activities performed by the testing team to ensure the quality of the software or the product.
* STLC deals only with the testing phases.
* STLC starts as soon as requirements are defined or SRD (Software Requirement Document) is shared by stakeholders.
* In the early stage of STLC, while the software or the product is developing, the tester can analyse and define the scope of testing, entry and exit criteria and also the Test Cases. It helps to reduce the test cycle time along with better quality.

**STLC Phases**

There are following six major phases in every Software Testing Life Cycle Model (STLC Model):

1. Requirement Analysis
2. Test Planning
3. Test case development
4. Test Environment setup
5. Test Execution
6. Test Cycle closure

STLC Model Phases

Each of these stages has a definite Entry and Exit criteria, Activities & Deliverables associated with it.

### 

* **Entry Criteria:** Entry Criteria gives the prerequisite items that must be completed before testing can begin.
* **Exit Criteria:** Exit Criteria defines the items that must be completed before testing can be concluded

You have Entry and Exit Criteria for all levels in the Software Testing Life Cycle (STLC)

**4.) Sanity and Smoke Testing.**

## Smoke Testing

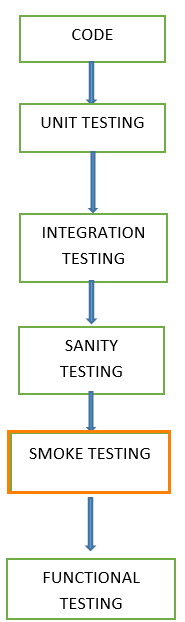
**Smoke Testing** is a software testing technique performed post software build to verify that the critical functionalities of software are working fine. It is executed before any detailed functional or regression tests are executed. The main purpose of smoke testing is to reject a software application with defects so that QA team does not waste time testing broken software application.

## What is Sanity Testing?

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in a more rigorous testing.

**KEY DIFFERENCE**

* Smoke Testing has a goal to verify “stability” whereas Sanity Testing has a goal to verify “rationality”.
* Smoke Testing is done by both developers or testers whereas Sanity Testing is done by testers.
* Smoke Testing verifies the critical functionalities of the system whereas Sanity Testing verifies the new functionality like bug fixes.
* Smoke testing is a subset of acceptance testing whereas Sanity testing is a subset of Regression Testing.
* Smoke testing is documented or scripted whereas Sanity testing isn’t.
* Smoke testing verifies the entire system from end to end whereas Sanity Testing verifies only a particular component.



X—Mapping Docs, --Testing Done—Target—30 TC---6

Y—Mapping, Req---Testing---40 TCs--10

**5.Regression Testing**

* DWH implementation is not a one-time job. This requires continual upgradation as the operational data store change and also periodic data loads are necessary.
* A well-defined base lined regression suit ensures existing functionality remains unchanged each time such a change happens.
* For Regression Testing we don’t write Test cases whatever test cases will be there will use that.
* Test cases for Regression testing can be considered as per business risk or importance. This helps in deciding which Test cases to be run for each of the new release or changes to the Up/Down stream application.
* A simple but effective strategy to Regression Testing of basic functionality is::
  + 1. Take backup of previous version run test result.
    2. Results for successful runs of the new code.
    3. And compare new test results with previous runs test results.
* When doing a regression test, It is much quicker to compare new results with previous execution results, then to do an entire data validation again for new code.

## 6.Test Strategy

* **Test Strategy** in software testing is defined as a set of guiding principles that determines the test design & regulates how the software testing process will be done.
* The objective of the[Test Strategy](https://www.guru99.com/how-to-create-test-strategy-document.html)is to provide a systematic approach to the software testing process in order to ensure the quality, traceability, reliability and better planning.
* **Test Strategy Document** is a well-described document in software testing which clearly defines the exact software testing approach and testing objectives of the software application.

## 7.Test Plan

* A Test Plan is a document that describes the objectives, scope, approach and focus of a software testing effort. The process of preparing a test plan is a useful way to think through the efforts needed to validate the software product.
* The completed test plan document will help people outside the test group understand the way and how of product validation.
* A **Test Plan** is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product.
* Test Plan helps us determine the effort needed to validate the quality of the application under test.
* The test plan serves as a blueprint to conduct software testing activities as a defined process, which is minutely monitored and controlled by the test manager.

**Test Plan Document Contents:**

* Introduction
* Scope of project
* Features not to be tested (Out of scope)
* Test Strategy
* Test Environment
* Entry Criteria
* Exit Criteria
* Release Deliverables
* Resource details
* Training Requirements
* Risks etc.

## Difference Between Test Strategy and Test Plan

|  |  |
| --- | --- |
| **Test Plan** | **Test Strategy** |
| * A test plan for software project can be defined as a document that defines the scope, objective, approach and emphasis on a software testing effort | * Test strategy is a set of guidelines that explains test design and determines how testing needs to be done |
| * Components of Test plan include- Test plan id, features to be tested, test techniques, testing tasks, features pass or fail criteria, test deliverables, responsibilities, and schedule, etc. | * Components of Test strategy includes- objectives and scope, documentation formats, test processes, team reporting structure, client communication strategy, etc. |
| * Test plan is carried out by a testing manager or lead that describes how to test, when to test, who will test and what to test | * A test strategy is carried out by the project manager. * It says what type of technique to follow and which module to test |
| * Test plan narrates about the specification | * Test strategy narrates about the general approaches |
| * Test plan can change | * Test strategy cannot be changed |
| * Test planning is done to determine possible issues and dependencies in order to identify the risks. | * It is a long-term plan of action.You can abstract information that is not project specific and put it into test approach |

**>>Bug/Defect Life Cycle<<**

**>>New:**

Whenever the defect is found for the first time the Test engineer will set the status as new.

**>>Open::**

Whenever the developer accepts the raised defect then he will set the status as open.

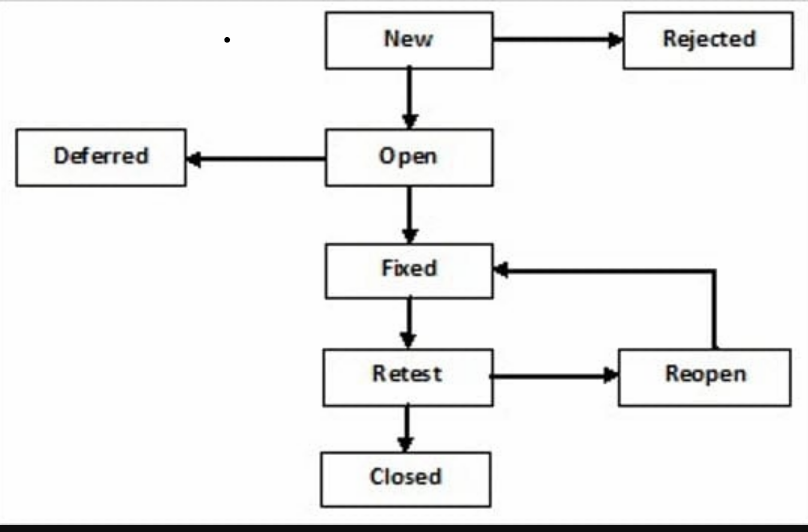
**>>Re-open and Closed::** Whenever the defects are rectified, Next Build is released to the Testing Department then the test engineers will check whether the defects are rectified properly or Not. If the defect is rectified properly then the Test engineer will set the status as ‘Closed’. If the defect is not rectified properly then the test engineer will set the status as “Re-Open.”

**>>Differed::**

Whenever the defect is arises test engineer will assign to the developer, If the defect is not going to fix in the current release and will be fixing in the future phase then the Test engineer will change the status as “Differed”.

>> **Rejected**:: Whenever the developer is confirmed it is not at all a defect then he will set the status of defect as “Reject”.

>>**Fixed**:: Whenever the Developer fixes the raised defect then he will change the status as “Fixed”.



>> **Severity**:: The Seriousness of the defect is defined in Terms of “Severity”.

Severity is Classified into four types.:

* 1. Critical (S1)
  2. Serious/High (S2)
  3. Medium (S3)
  4. Minimal/Low (S4).

### **Priority:** **What Is Priority?**

Priority, as the name suggests, is about prioritizing a defect based on business needs and severity of the defect. Priority signifies the importance or urgency of fixing a defect.

While opening a defect, the tester generally assigns the priority initially as he views the product from the end-user perspective. In line with these, there are different levels:

**Broadly, Priority of the defects can be classified as follows:**

#### **Priority #1) Immediate/Critical (P1)**

This has to be fixed immediately within 24 hours. This generally occurs in cases when an entire functionality is blocked and no testing can proceed as a result of this. Or in certain other cases if there are significant memory leaks, then generally the defect is classified as a priority -1 meaning the program/ feature is unusable in the current state.

Any defect that needs immediate attention which impacts the testing process will be classified under the immediate category

All the **Critical severity**defects fall under this category (unless re-prioritized by business/stakeholders)

#### **Priority #2) High (P2)**

Once the critical defects have been fixed, a defect having this priority is the next candidate which has to be fixed for any test activity to match the “exit” criteria. Normally when a feature is not usable as it’s supposed to be, due to a program defect, or that new code has to be written or sometimes even because some environmental problem has to be handled through the code, a defect may qualify for a priority 2.

This is the defect or issue which should be resolved before the release is made. These defects should be resolved once the Critical issues are solved.

All the **Major** **severity** defects fall into this category.

#### **Priority #3) Medium (P3)**

A defect with this priority must be in contention to be fixed as it could also deal with functionality issues which are not as per expectation. Sometimes even cosmetic errors such as expecting the right error message during the failure could qualify to be a priority 3 defect.

This defect should be resolved after all the serious bugs are fixed.

Once the Critical and the High priority bugs are done, we can go for the medium priority bugs.

All the**Minor** **severity** defects fall into this category.

#### **Priority #4) Low (P4)**

A defect with low priority indicates that there is definitely an issue, but it doesn’t have to be fixed to match the “exit” criteria. However, this must be fixed before the GA is done. Typically, some typing errors or even cosmetic errors as discussed previously could be categorized here.

Sometimes defects with priority low are also opened to suggest some enhancements in the existing design or a request to implement a small feature to enhance user experience.

This defect can be resolved in the future and does not need any immediate attention and the **Low severity** defects fall into this category.

As already discussed priority determines how quickly the defect turnaround time must be. If there are multiple defects, the priority decides which defect has to be fixed and verified immediately versus which defect can be fixed a bit later.

**S1/P1---When your Testing will blocked**

**S2/P2---When Data Mismatch or Count mismatch,**

**S3/P3-- When Data Mismatch or Count mismatch,**

**S4/P4—When Datatype/Length is not matching as per mapping docs.**



Q. **What are the Defect Contents?**

Complete information such that the developers can understand the bug. Get an idea of its severity and reproduce if it necessary. Defect will contain the following contents.

* Bug ID
* Current Defect Status
* Defect Title (Summary)
* Description of Steps needed to reproduce the Bug.
* Severity & Priority
* Tester Name
* Attachments
* Expected Results
* Actual Result

**>>What is Test Cases<<**

**A Test Cases is a document that describe an input, action, or event and its expected result, in order to determine if a functionality of an application is working correctly.**

* **How Can you say that you have done good testing?**
* Test cases completed with certain percentage passed
* Coverage of functionality reaches a specified point
* Bug rate falls below a certain level.
* **What are the criteria for writing test cases?**
* A basic criteria for writing test cases is to understand the requirements thoroughly. Once your understanding for requirement is clear, you can start writing test cases.
* Another criteria is your test case format should be ready.
* You should document all your requirements properly and make sure that for each and every requirement there must be at least one test cases is covered.

**Q. When the Testing Process Starts?**

>> The Testing Process Starts after the requirement analysis: Writing Test cases, Reviewing Test Cases Etc. and The Actual Testing starts after Coding. Execution of the Test Cases**.**

**Q. What is BRD?**

BRD Stands for Business Requirement Document. This document specifies the needs and the requirements of the customer.

**Q. What is Black Box Testing?**

Ans: Black Box Testing is Functional Testing, Not based on any knowledge of Internal software design or Code. Black box testing is based on requirements and functionality.

**Q. What is White box Testing?**

Ans: White box testing is based on knowledge of the Internal logic of an application’s code. Tests are based on coverage of code statements, branches, path and conditions.

**Q. What are the different levels of Testing?**

The different level of Testing are:

* Functional Testing
* Integration Testing
* System Testing
* Regression Testing
* UAT Testing

**Q. What is Unit Testing?**

Unit Testing focuses verification effort on the smallest unit of software module. The Unit Testing is normally White Box oriented. Unit Testing is done by programmers to verify the written code.

**Q. What is Sanity Testing?**

Sanity Testing is a casual Testing. It is performed whenever new build(new Code) Released to Test environment to prove the application is functioning according to specifications before starts the functional Testing.

This Normally includes a set of Core Tests of Basic Functionality to demonstrate connectivity to the database, application servers etc are set correctly.

**Q. What is Functional Testing?**

Functional Testing is Black Box Testing accommodates to Functional requirements of an application. Test Engineers should perform functional Testing.

**Q. What is Integration Testing?**

The Purpose of Integration Testing is to ensure distinct components (Upstream and Downstream) of the application still work in according to customer requirements.

Integration Testing Test Cases are developed with the express purpose of Testing the Interfaces between components (Upstream and Downstream).

This activity is carried out by Test Team.

**Q. What is System Testing?**

System Testing performed by Test Team and at the start of the system testing the complete system is configured like production environment (Required software, Hardware, etc..)

The purpose of System testing is to validate an application’s accuracy and completeness in performing the functions as designed. System Testing starts after completion of Integration Testing.

**Q. What is End to End Testing?**

End to End Testing is similar to system Testing. It is the process of Testing the complete process from end 2 end.

This to ensure that the overall process flows as expected, and the system components integrates together correctly and the right information is passed between the applications (Upstream, downstream and Report).

**Q. What is Re-testing?**

The defect once raised and as the development team fixes the bug, it has to undergo the testing process again to the same functionality to ensure that the fixed defect is working as expected, this is called Re-Testing.

**Q. How can it be known when to stop Testing?**

1. Deadlines (Release Deadlines, Testing Deadlines
2. Test cases Completed with certain percentage passed
3. Test Budget spent and completed
4. Coverage of Functionality/requirements reaches a specified point.
5. Bug rate falls below a certain level.

**Q. What is Traceability Matrix?**

Traceability Matrix is nothing but a mapping between the requirement and Test cases. How many test cases written for particular Requirement and all the test cases for specified requirement?

**Q. What is Database Black Box Testing?**

Black Box Testing is a software testing approach that involves testing the functions of software applications without knowing the internal code structure, implementation details, or internal routes. Black Box Testing is a type of software testing that focuses on the input and output of software applications and is totally driven by software requirements and specifications. Behavioral testing is another name for it.

**>>>Step to reproduce the defect:::**