**\*\*\*\*\*\*\*\*\*\*\*\*Defect Life Cycle Notes\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*Software Testing:-** It is the process of comparison between the expected result & actual result. In this, correctness & completeness of the software also checked. It is also done to maintain & achieve the desired quality of software.

**\*Attributes of Software Testing:-**

1. **Reliability**-Application can run failure free, until its life is end.

2. **Scalability**-Impact of software after load, to check the performance of application.

3. **Portability** - Ability of software to be transferred from one machine / system to another.

4. **Reusability** - Reuse of existing code by doing some changes within the software product development process.

5. **Usability**-How application is user-friendly?

**\*Definition of Manual Testing :-** Manual testing is a software testing process in which test cases are executed manually without using any automated tool. All test cases executed by the tester manually according to the end user's perspective.

**\*Automation Testing** :- Automation Testing is a process of converting any manual test cases into the test scripts with the help of automation tools or any programming languages. Using a test automation tool, it's possible to record this test suite and re-play/playback it as required. Once the test suite is automated, no human intervention is required.

**\*Functional Testing**:-Functional Testing is a type of software testing which is used to verify the functionality of the software application, whether the function is working according to the requirement specification. Functional aspects are unit, integration, system, and user acceptance testing.

**\*Non-Functional Testing** :- Non-Functional Testing is a type of Software testing to check non-functional aspects (performance, usability, reliability) of a software application. It is designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing.

\*\*\***API(Application Programming Interface):-API** is a formalized set of software calls and routines that can be referenced by an application program to access supporting system or network services.

# \*\*\*\*\*What is an API? (Application Programming Interface):-

API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other. Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you’re using an API.

## \*\*\*What Is an Example of an API?

When you use an application on your mobile phone, the application connects to the Internet and sends data to a server. The server then retrieves that data, interprets it, performs the necessary actions and sends it back to your phone. The application then interprets that data and presents you with the information you wanted in a readable way. This is what an API is - all of this happens via API.

Imagine you’re sitting at a table in a restaurant with a menu of choices to order from. The kitchen is the part of the “system” that will prepare your order. What is missing is the critical link to communicate your order to the kitchen and deliver your food back to your table. That’s where the waiter or API comes in. The waiter is the messenger – or API – that takes your request or order and tells the kitchen – the system – what to do. Then the waiter delivers the response back to you; in this case, it is the food.

Here is a real-life API example. You may be familiar with the process of searching flights online. Just like the restaurant, you have a variety of options to choose from, including different cities, departure and return dates, and more. Let us imagine that you’re booking you are flight on an airline website. You choose a departure city and date, a return city and date, cabin class, as well as other variables. In order to book your flight, you interact with the airline’s website to access their database and see if any seats are available on those dates and what the costs might be.

However, what if you are not using the airline’s website–a channel that has direct access to the information? What if you are using an online travel service, such as Kayak or Expedia, which aggregates information from a number of airline databases?

The travel service, in this case, interacts with the airline’s API. The API is the interface that, like your helpful waiter, can be asked by that online travel service to get information from the airline’s database to book seats, baggage options, etc. The API then takes the airline’s response to your request and delivers it right back to the online travel service, which then shows you the most updated, relevant information.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

\*\*\***Graphical User-interface Testing or GUI testing :-GUI Testing** is **a process of testing the user interface of an application**. GUI testing is done to check the functionality and usability of design elements as a user for an application under test.

\*\*\*In **GUI Testing** following points are considered:-

**1.The same font type and size is used for all over the application.**

**2.Colors and style guide of the product confirm the specifications.**

**3.Input fields are of proper size.**

**4.Navigation is intuitively clear.**

**5.All icons are designed in one style.**

**6.Interface contains no visual and dialogbox consistencies.**

\*\***\*N+1 testing:-** The variation of regression testing is represented as N+1. In this technique, the testing is performed in multiple cycles in which errors found in test cycle ‘N’ are resolved and re-tested in test cycle N+1.

**\*\*\*Fuzz testing:-** Fuzz testing is used to detect security loopholes and coding errors in software. In this technique, random data is added to the system in an attempt to crash the system.

|  |  |
| --- | --- |
| **Functional Testing** | **Non-Functional Testing** |
| Unit testing, Integration testing, System testing, Acceptance testing,  Regression testing, Smoke testing, Sanity testing, White Box testing. | Load testing, Performance testing, Localization testing, Recovery testing Internationalization testing, Security testing, Compatibility testing, Volume testing, Stress testing. |

### ****\*\*\*\*advantages of manual testing:-****

**1.**It is a cheaper way of testing when compared to automated testing.

**2**.Analysis of product from the point of view of the end-user is possible only with manual testing.

**3.**GUI testing can be done more accurately with the help of manual testing as visual accessibility and preferences are difficult to automate.

**4.**Easy to learn for new people who have just entered into testing.

**5.**It is highly suitable for short-term projects when test-scripts are not going to be repeated and reused for thousands of times.

**6.**Best suited when the project is at the early stages of its development.

**7.**Highly reliable, since automated tests can contain errors and missed bugs.

### ****\*\*\*\*\*\*drawbacks of manual testing:-****

**1**.Highly susceptible to human error and are risky.

**2**.Test types like load testing and performance testing are not possible manually.

**3.**Regression tests are really time-consuming if they are done manually.

**4**.Scope of manual testing is very limited when compared to automation testing.

**5.**Not suitable in very large organizations and time-bounded projects.

**6.**The cost adds up, so it’s more expensive to test manually in the long run.

# \*Software Development Life Cycle (SDLC):-

**Phases of software development cycle:-**

**1.[Requirement Phase](https://www.javatpoint.com/software-development-life-cycle" \l "requirement-phase)**:- The client states requirements, specifications, expectations, and any other special requirement related to the product or software are gathered by the business manager or project manager or analyst.

**2.[Design Phase](https://www.javatpoint.com/software-development-life-cycle" \l "design-phase)**:- In this phase, the logical designing of the system is converted into physical designing.

There are several techniques and tools, such as data flow diagrams, flowcharts, decision tables, and decision trees, Data dictionary, and the structured dictionary are used for describing the system design.

**3.[Build /Development Phase](https://www.javatpoint.com/software-development-life-cycle" \l "build-development-phase):-** In this phase, implement the design into the development of a software system. Coding starts by the team of developers according to the design discussed in the previous phase and according to the requirements of the client.

Front-end developers:-Develop easy and attractive GUI and necessary interfaces to interact with back-end operations.

back-end developers:-Develop back-end coding according to the required operations.

**4.[Testing Phase](https://www.javatpoint.com/software-development-life-cycle" \l "testing-phase):**-After getting the developed GUI and back-end combination, it is tested against the requirements stated in the requirement phase. Testing determines whether the software is actually giving the result as per the requirements or not.

Integration testing, unit testing, acceptance testing, and system testing are carried out. If there are any defects in the software or it is not working as per expectations, then the testing team gives information to the development team.

**5.[Deployment/ Deliver Phase](https://www.javatpoint.com/software-development-life-cycle" \l "deployment-deliver-phase):-** When software testing is completed with a satisfying result, it is delivered to the customer for their use. As soon as customers receive the product, they are recommended first to do the beta testing. If it is having a severe issue, then the development team solves it in a short time; otherwise, if it is less severe, then it will wait for the next version.

After the solution of all types of bugs and changes, the software finally deployed to the end-user.

**6.[Maintenance](https://www.javatpoint.com/software-development-life-cycle" \l "maintenance):-** The maintenance phase is the last and long-lasting phase of SDLC because it is the process which continues until the software's life cycle comes to an end. This phase also includes making changes in hardware and software to maintain its operational effectiveness like to improve its performance, enhance security features and according to customer's requirements with upcoming time. This process to take care of product time to time is called maintenance.

**\*\*\*\*\*Software testing life cycle contains the following steps:**

1. [**Requirement Analysis**](https://www.javatpoint.com/software-testing-life-cycle#requirement-analysis)
2. [**Test Plan Creation**](https://www.javatpoint.com/software-testing-life-cycle#test-plan-creation)
3. [**Environment setup**](https://www.javatpoint.com/software-testing-life-cycle#environment-setup)
4. [**Test case Execution**](https://www.javatpoint.com/software-testing-life-cycle#test-case-execution)
5. [**Defect Logging**](https://www.javatpoint.com/software-testing-life-cycle#defect-logging)
6. [**Test Cycle Closure**](https://www.javatpoint.com/software-testing-life-cycle#test-cycle-closure)

**\*Parameters of** **Non-Functional Testing:-Efficiency, (Performance, Loading capacity, Security, Portability, Accountability, Reliability).**

**Bug** – A bug is a fault in the software that’s detected during testing time. They occur because of some coding error and leads a program to malfunction. They may also lead to a functional issue in the product. These are fatal errors that could block a functionality, results in a crash, or cause performance bottlenecks.

**Defect** – A defect is a variance between expected results and actual results, detected by the developer after the product goes live. The defect is an error found AFTER the application goes into production. In simple terms, it refers to several troubles with the software products, with its external behavior, or with its internal features.

**Error** – An error is a mistake, misunderstanding, or misconception, on the part of a software developer. The category of developers includes software engineers, programmers, analysts, and testers. For example, a developer may misunderstand a design notation, or a programmer might type a variable name incorrectly – leads to an error. An error normally arises in software , it leads to a change the functionality of the program.

In simple terms when a defect reaches the end customer, it is called a failure while the defect is identified internally and resolved; then it is referred to as a defect.

\***Bug:**-**Defect approved / accepted by developer, then defect is bug.**

**\*Defect-Tester point of view.**

**\*Bug-Developer point of view.**

**\*Deferred Defect/Postpone**:-If a present bug is not of a prime / high priority & then it is expected to get fixed in the next release.

**\*Severity**:-Impact of the defect on an application. **Severity** decided by **Test Engineers/Testers.**

**\*3 types of Severity**:-**High, Medium, Low[or Blocker**(Major feature is not working, Major flow is not working) ,**Critical**(Main functionality is not working),**Major, Minor].**

**\*Priority:-**Importance of fixing the defect/How fast the defect is fixed. **Priority** decided by **Developers, Product Managers.**

**\*3 types of Priority: - Urgent, High, Medium, Low.**

|  |  |  |  |
| --- | --- | --- | --- |
| **High Severity-**  **High Priority** | **High Severity- Low Priority** | **Low Severity- High Priority** | **Low Severity-Low Priority** |
| **1.** If Users are Unable to Login for application with Valid Credentials. | **1.** In Flight reservation Application, No flight Reserved. | **1.** In any application Company Name Spelling Mistake. | **1.** Spelling Mistake in a webpage, frequently navigated by user. |
| **2.** ATM Machine- Money is not received & Msg is coming. | **2.** If the application crashes, after multiple use of any | **2.** On application Logo Mistake or Logo color. | **2.** The caption on an image is written in wrong font. |

## \*\*\*Severity and Priority Real-time Examples:-

The priority and severity are combined in four different ways to determine which defect needs immediate attention and which one the least.  Let’s look at some real-time examples to make this concept even clearer.

|  |  |  |  |
| --- | --- | --- | --- |
| **High Severity & High Priority** | **High Severity & Low Priority** | **Low Severity & High Priority** | **Low Severity & Low Priority** |
| You log in to your [amazon.com](https://www.amazon.com/) account, add items to the cart and click the “Proceed to Checkout” button. You make the payment and the system crashes. This defect makes the whole buying functionality unusable and so the severity is high. | Suppose the tester clicks on the “Privacy Notice” hyperlink at the bottom of the amazon.com home page and the page is not displayed. This defect will be of high severity because the functionality is not working. The priority is low because people do not normally spend time reading the privacy notice. | Suppose, that in the amazon.com website, the logo is displayed as ”amazn.com” with the letter “o” missing. This defect does not affect the buying/selling or any other functionality in any way.  So, the severity of this defect is low. But, a mistake in the company logo affects the brand identity and impacts the user experience. So, the defect is of high priority. | Suppose the tester clicks on the “Conditions of Use” hyperlink at the bottom of the amazon.com homepage. If there is an alignment issue in the text displayed or if there is a spelling mistake in the content displayed, the defect is said to be of low priority because people rarely read this page and it does not impact the user experience. The severity is also low because the functionality of the application is not affected. |

|  |  |  |  |
| --- | --- | --- | --- |
| ****High Severity and High Priority Examples**** | ****High Severity and Low Priority Examples**** | ****High Priority and Low Severity Examples**** | ****Low Priority and Low Severity Examples**** |
| 1.The products added to the cart of an e-commerce website are not visible on the payment page. | 1. If the application is crashing on passing very large input for processing (which is very rarely done). | 1. The logo of the company’s welcome page is distorted (misrepresented). | 1. A spelling mistake on the page of the site which is not frequently visited. |
| 2. The login functionality (button) of the application is not working. | 2. There are some buttons on the website which are overlapping. Although clickable, are creating a fuss (unnecessary interest). | 2. The action buttons are not visually appealing or the information on the page appears hazy (not clear). | 2. The color of any text does not match the theme of the website. |

**\*\*\*\***

**1.High Priority, High Severity :-** An error which occurs on the basic functionality of the application and will not allow the user to use the system(E.g. user is not able to login to the application).

**2.High Severity, Low Priority :-** Web page not found when user clicks on a link.

**3.High Priority, Low Severity :-** If the company name is misspelled in the home page of the website, then the priority is high and severity is low to fix it.

**4.Low Priority, Low Severity :-** Any cosmetic or spelling issues which is within a paragraph or in the report.

**\*Defect/Bug life cycle:-**

We have some different status of bugs like **new/open, assigned, fix, re-open, re-test and closed**. (4 types of defect (Open)-🡪Duplicate, Not a Defect, Deferred/Postpone, Rejected)

**After defect Fixed & Closed :- We have to make New Defect Report.**

**After defect Re-Open:- We have to make New Defect Report.**

* **New:** When a new defect is logged and posted for the first time. It is assigned a status as NEW.
* **Assigned:** Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team.
* **Open**: The developer starts analyzing and works on the defect fix.
* **Fixed**: When a developer makes a necessary code change and verifies the change, he or she can make bug status as "Fixed."
* **Pending retest**: Once the defect is fixed the developer gives a particular code for retesting the code to the tester. Since the software testing remains pending from the testers end, the status assigned is "pending retest."
* **Retest**: Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to "Re-test."

**\*Whom to assign the bug:-**

The bug can be assigned to the following:

* **Developers**
* **Developer lead**
* **Test lead**

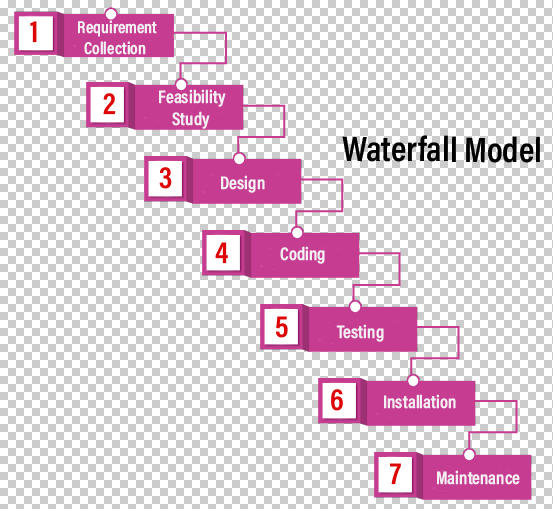
[](https://www.guru99.com/images/1-2015/012715_0802_BugLifeCycl1.png)

**\*Waterfall Model:-**

It is the first approach and the basic model used in software development. It is a simple model that is easy to use as well as understand. The execution happens in the sequence order, which means that the outcome of the one-stage is equal to the input of another stage. That's why it is also known as the Linear-sequential life cycle model.

To avoid the overlapping issues of the multiple phases, every stage should be completed before moving to the next stage. Each stage of the waterfall model involves the deliverable of the previous stage, like requirements, are transferred to the design phase, design moved to development, and so on. When we have the Life critical (hospital application) and Machine critical (Military project), we will widely use the waterfall model.

This Application/Software is handover to the customer, when complete software is develop. Each phase is depend on the previous phase.



## In between development of application/software , we cannot add any requirement of customer. For changes in application ,we have to develop application again from initial phase.

## Example :- **Human Resource Management [HRM], Supply Chain Management System, Customer Relationship Management [CRM], and Retail Chains.**

# \*Spiral Model:-

The biggest problem we face in the waterfall model is that taking a long duration to complete the product, and the software became outdated. To solve this problem, we have a new approach, which is known as the Spiral model. The spiral model is also known as the cyclic model.

In this model, we create the application module by module and handed over to the customer so that they can start using the application at a very early stage. And we prepare this model only when the module is dependent on each other. In this model, we develop the application in the stages because sometimes the client gives the requirements in between the process.

## Learn about the Top 10 Software Development Process ModelsPrototyping:-Dummy Working Application.

## 1. Planning :- Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.

## 2. Risk Analysis:-During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype (Dummy Working Application) is built for the best possible solution.

## 3. Engineering :- During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.

## 4. Evaluation:-In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.    \*\*\*\*\*\*\*Advantage:-

1. Flexible changes are allowed in spiral model.

## 2. The customer can use the application at an early stage also.

## \*V & V model/V model/Verification & Validation model:-

The entire V model executes in two-phase, the complete **review process** is done in the **verification phase**, and the whole **testing process** is done under the **validation phase**; that's why it is also known as **verification and validation** model.

In this model, development phase & testing phase simultaneously carried out.

Where the verification and validation process includes different stages:

## Verification:-Process based.

## Validation:-Product based.

## V-model\*V-model used for:-

## -For the large and complex application.

## - It is also used for the long term projects.

## \*[Static testing](https://www.javatpoint.com/static-testing) is a **verification** process,& ****Dynamic testing**** is a **validation** process, and together they help us to deliver a cost-effective quality Software product.

## \*Testing Techniques/Black Box Testing Techniques:-

## - While writing the test cases to achieve the maximum test coverage, testing techniques are used.

## - After applying Testing Technique, application is process-oriented rather than person-oriented.

**1.Error Guessing :-**

**-** It is **based on the experience of the test analyst**, where the tester uses the experience to guess the problematic areas of the software. to find the error.

**2. Equivalence Class Partitioning (ECP):-**

**-**In this method, input data is divided into partitions of valid and invalid values.

**-Age Range-18 to 25 ECP Value🡪(10,20,45)**

**1 to 17🡪Invalid 18 to 25->Valid 26<->Invalid**

**10 20 45**

- The principle of equivalence partitioning is, **test** **cases should be designed to cover each partition at least once.** Each value of every equal partition must exhibit the same behavior as other.

**3. Boundary Value Analysis [BVA]:-**

**-** It is used to test boundary values, **because the input values near the boundary have higher chances of error.**

**Age Range-18 to 25 BVA Value 🡪(10,20,45)**

**1 to 17🡪Invalid 18 to 25->Valid 26 <->Invalid**

**17, 18 ,19 ,24,25,26**

**-**  **BVA** contain the upper and lower limit of a variable.

**-** Testing of boundary values is done by making valid and invalid partitions. Invalid partitions are tested because testing of output in adverse condition is also essential.

**-** Tester develops test cases for both valid and invalid partitions to capture the behavior of the system on different input conditions.

**4. Decision Table Technique:-**

**-**A **Decision Table** is a tabular representation of inputs versus rules/cases/test conditions.

**-** Decision table testing is a software testing technique used to test system behavior for different input combinations.

**-** It is also called as a **Cause-Effect** **table** where Cause and effects are captured for better test coverage.

The condition is simple if the user provides correct username and password the user will be redirected to the homepage. If any of the input is wrong, an error message will be displayed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Conditions** | **Rule 1** | **Rule 2** | **Rule 3** | **Rule 4** |
| **Username (T/F)** | F | T | F | T |
| **Password (T/F)** | F | F | T | T |
| **Output (E/H)** | E | E | E | H |

Legend:

* T – Correct username/password
* F – Wrong username/password
* E – Error message is displayed
* H – Home screen is displayed

**\*Levels of Testing/Levels of Functional Testing:-**

**1. Unit Testing/Components Testing**:-

**-**A unit (unit or an individual component of the software application) is a single testable part of a software system and tested during the development phase of the application software. It is usually done by the developers**.**

**-**The aim behind unit testing is to validate unit components with its performance.

**2. Integration Testing:-**

-In this testing, units or individual components of the software are tested in a group. The goal of integration testing is to check the correctness of communication among all the modules.

-Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as **integration testing**.

**3. System Testing:-**

-System testing is a series of different type of tests with the purpose to examine the full working of an integrated software computer system against requirements. (with required hardware & Software integration).

-To check the end-to-end flow of an application or the software as a user is known as **System testing**.

-In this, we navigate (go through) all the necessary modules of an application and check if the end features or the end business works fine, and test the product as a whole system.

**4. Acceptance Testing:-**

**-** Acceptance testing is the most important phase of testing as this decides whether the client approves the application/software or not. It may involve functionality, usability, performance, and U.I of the application. It is also known as user acceptance testing (UAT), operational acceptance testing (OAT), and end-user testing. This is a type of testing done by users, customer.

## \*\*2 types of Acceptance Testing:-

## 1) Alpha testing/Factory testing :-This is performed by "developers" of the software application in "development side environment".

## 2) Beta testing/Field level testing :- This is performed by "real users" of the software application in "real environment". This testing helps to test products in “client side environment”.

## \*\*\*\*\*\*Requirement Traceability Matrix:-

## ****-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.

## \*\*\*\*\*\*\*\*\*\*\*Why RTM is Important?

-**RTM**:-It is a document which shows the cross reference relationship between requirements & test cases.

-The main agenda of every tester should be to understand the client’s requirement and make sure that the output product should be defect-free. To achieve this goal, every QA should understand the requirement thoroughly and create positive and negative test cases.

-Regression testing is performed for passed test cases while **Retesting is done only for failed test cases**. **Regression Testing** doesn't include **defect verification** whereas **Re-testing** includes **defect verification**. Regression testing is known as generic testing whereas Re-testing is planned testing.

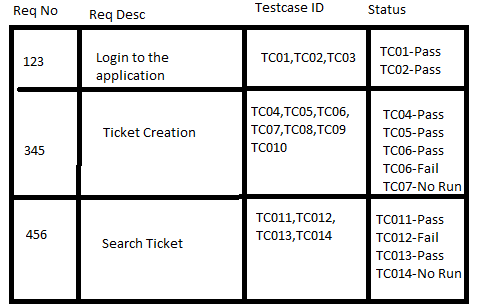
- In regression testing ,“Re-executing the pass test cases” can be done .In regression testing ,RTM can be prepared (Req. traceability matrix).This testing is generally carried out by software tester.

-Impact Analysis:-It suggest that, "How much regression need to be done"?

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

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

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

[](https://www.guru99.com/images/1/requirements-traceability-matrix.png)

Above is a sample requirement traceability matrix.

### \*\*\*White box Testing Techniques / different types of test coverage techniques are:-

🡪Different types of test coverage techniques include

* **Statement Coverage:** It verifies that each line of source code has been executed and tested.
* **Decision Coverage:** It ensures that every decision in the source code is executed and tested.
* **Path Coverage:** It ensures that every possible route through a given part of the code is executed and tested.

**\*\*\*\*\*\*\* Types of Functional Testing:-**

## \*Regression Testing(By Re-execution passed test cases):- **Regression** Testing is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features.

## \*Retesting(By Re-execution failed test cases):-**Retesting** is a process to check specific test cases that are found with bugs in the final execution.

## \*\*\*\*Re-testing ensures the original fault has been removed; Regression testing looks for unexpected side effects.

## \*Smoke Testing/****Build Verification Testing**/**Confidence Testing** .**:-Smoke Testing is a testing in which newly complete build (initial build) software/application (received by tester at first time) checking the all basic functionality of software to verify whether the application is stable for further testing or not. Smoke testing is performed by both developers and testers.

## \*Sanity testing/Sub-Regression 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. Sanity testing is performed by testers alone.

## \*Smoke testing is done on initial/fresh/newly build application & Sanity Testing is done on matured build.

**\*Exploratory Testing :-** Exploratory **Testing** is a type of software testing where Test cases are not created in advance but testers check system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a "thinking" activity.

Exploratory Testing is widely used in Agile models and is all about discovery, investigation, and learning. It emphasizes personal freedom and responsibility of the individual tester.

Simultaneous test design & execution against an application is called **“Exploratory Testing”.** In this testing, the tester uses his domain knowledge & testing experience to predict where & under what conditions the system might behave unexpectedly.

**“Exploratory testing”** is performed as a final check before the software is released. It is a complementary activity to automated regression testing.

**\*User Acceptance Testing/Acceptance Testing :-** **User Acceptance Testing (UAT)**is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.

**\*\*Risk Based Testing:-The Risk-based testing** is a testing strategy that is based on prioritizing tests by risks .It is based on a detailed risk analysis approach which categorizes the risks by their priority .Highest priority risks are resolved first.

**\*\*\*Agile testing:-Agile testing** is a testing practice that uses agile methodologies i.e. follow test-first design paradigm.

**\*\*\*Cross Browser testing**:- It is a type of non-functional testing that lets you check whether your website works as intended when accessed through:

**1.Different Browser-OS combinations** i.e., on popular browsers like Firefox, Chrome, Edge, Safari—on any of the popular operating systems like Windows, macOS, iOS and Android.

**2.Different devices** i.e., users can view and interact with your website on popular devices—smartphones, tablets, desktops and laptops etc.

**3.Assistive Tools** i.e., the website is compatible with assistive technologies like screen readers for individuals who are differently abled.

## \*Adhoc testing/Monkey testing/Gorilla testing :- ****Adhoc Testing**** is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage. Ad hoc testing is done randomly and it is usually an unplanned activity which does not follow any documentation and test design techniques to create test cases.

## Adhoc Testing is a testing phase where the tester tries to “break” the system by randomly trying the system’s functionality.

### Q) How much testing is enough?

🡪It is depends on the risk for your industry, contract and special requirements.

### Q). When should testing be stopped?

🡪It depends on the risks for the system being tested. There are some criteria based on which you can stop testing like as follows:-

1. **Deadlines (Testing, Release)**
2. **Test budget has been depleted(decreased ,reduced)**
3. **Bug rate fall below a certain level**
4. **Test cases completed with certain percentage passed**
5. **Alpha or beta periods for testing ends**
6. **Coverage of code, functionality or requirements are met to a specified point.**

|  |  |
| --- | --- |
| Adhoc Testing | Exploratory Testing |
| 1. **Adhoc Testing** is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage. | 1. **Exploratory Testing**  is a type of software testing where Test cases are not created in advance but testers check system on the fly. |
| 2. This testing requires no documentation/ planning /process to be followed. | 2. This testing followed simulta---neous process of test design & test execution all done at the same time. |

|  |  |
| --- | --- |
| **Static testing** | **Dynamic testing** |
| In this testing, check the application without executing the code. It is a **Verification**Process. | In this testing, check the application by executing the code. It is a **Validation** Process. |
| Static testing includes activities like Inspection, Review, Walkthrough, etc. | Dynamic testing includes functional and non-functional testing such as usability testing, integration testing, System testing & user acceptance testing. |
| Static testing is used to prevent defects. | Dynamic testing is used to find and fix the defects. |
| Static testing is a more cost-effective process. | Dynamic testing is a less cost-effective process. |
| Under static testing, we can perform the statement coverage testing and structural testing. | Equivalence Partitioning & Boundary Value Analysis technique are performed under dynamic testing. |

## \*Localization testing: - **Localization Testing** is a software testing technique in which the behavior of a software is tested for a specific region, locale or culture. It is the process of customizing the software as per the targeted language and country. Language engineers & translator are help to convert the language.

## \*Globalization/Internationalization testing: - **Globalization Testing** is a software testing method used to ensure that the software application can function in any culture or locale (language, territory or code page) by testing the software functionalities using each type of international input possible.

## -It means test that application is adoptable globally across the world, without need of reprogramming. The purpose of Globalization testing is to ensure that software can be used internationally or worldwide.

## \*Security testing:-It is a type of testing that uncovers vulnerabilities, threats*, risks in a software application and* prevents malicious attacks from intruders.

## Following points are considered in Security Testing:-

## 1. Authentication

## 2. Authorization

## 3. Vulnerability Scanning

## 4. Security Scanning

## 5.Penetration testing

## 6.Risk Assessment

## 7.Security Auditing

## 8.Ethical Hacking.

## \*Performance testing :- **Performance Testing** is a software testing process used for testing the speed, response time, stability, reliability, scalability and resource usage of a software application under particular workload.

## Generally these 3 things:-Load, Speed, Response Time, Stability can be checked in Performance testing.

## \*\*\*\*\*\*\*Types of Performance testing:-

## 1. Load Testing:- **Testing the application with peak load is called Load Testing. It is conducted to understand the behavior of application under a specific expected load. (**Peak Load:-Total no. of users given by customer**)**

## ****2. Stress testing**:-Testing the application with beyond the peak load is called StressTesting. Main purpose behind this testing is to determine the degradation point (Saturation point) & Crash Point (Breaking point) of application when the application is loaded with beyond the peak load.**

## ****3.Endurance testing/Soak testing**:-Testing the application with a significant load extended over a significant period of time, to discover how the system behave under sustained use is called Endurance testing.**

## ****4. Volume testing**:-Testing the application w.r.to. database size is called Volume testing. Main purpose of this testing is to determine the performance of application with increasing volumes of data in database.**

## **\***Configuration Testing**:-Testing application with different possible hardware configuration is called Configuration Testing.**

## **\***Compatibility testing/Multiplatform testing:-**It is testing in which how well software performs in a particular hardware, software, OS or network environment.**

## ****\*User Interface testing (GUI testing)**:-In this testing, check that how user friendly the application is. In GUI testing, Appropriate input, Correct messages, Spelling check, Graphic check, Meaningful error messages, Meaningful help documents, Accuracy of displayed data all these things are considered.**

## \*Stability testing :- **It is a** type of non-functional software testing performed to measure efficiency and ability of a software application to continuously function over a long period of time.

## \* Accessibility testing:-It is defined as a type of testing performed to ensure that the application being tested is usable by (disable person, handicapped person) people with disabilities like hearing, color blindness, old age and other disadvantaged groups. It is a subset of Usability Testing.

## \*Scalability Testing:-It is a testing that ensures that the system works well in proportion to the growing demands of the end users. Following are the attributes checked during this testing:-Response time, Throughput, No.of users required for performance test ,Threshold Load ,CPU usage, Memory usage, Network usage.

## \*\*\*Authentication:- User’s Identity checked.

## \*\*\*Authorization:- Level of Access/ User’s Identity checked.

## \*Difference of Authentication & Authorization:-

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Authentication** | **Authorization** |
| **1.** | In authentication process, the identity of users are checked for providing the access to the system. | While in authorization process, person’s or user’s authorities are checked for accessing the resources. |
| **2.** | In authentication process, users or persons are verified. | While in this process, users or persons are validated. |
| **3.** | It is done before the authorization process. | While this process is done after the authentication process. |
| **4.** | It needs usually user’s login details. | While it needs user’s privilege or security levels. |

|  |
| --- |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Sanity Testing** | **Regression Testing** |
| 1 | Sanity Testing is performed to check the stability of new functionality or code changes in the existing build. | Regression testing is performed to check the stability of all areas impacted by any functionality change or code change. |
| 2 | Sanity Testing is part of Regression Testing. | Regression Testing is independent testing. |
| 3 | It is executed before Regression Testing and after [Smoke Testing](https://www.geeksforgeeks.org/smoke-testing-software-testing/). | It is executed based on the project and availability of resources, manpower and time. |
| 4 | Sanity Testing is considered as a Surface Level Testing. | Regression Testing is not considered as a Surface Level Testing. |
| 5 | It examines few functionality of the software. | It examines extended mostly all functionality of the software. |
| 6 | Sanity Testing does not use scripts. | Regression Testing uses Scripts. |
| 7 | Sanity Testing is often carried out manually. | Regression Testing is often preferred to continue with automation. |
| 8 | Performing Sanity Testing increases the product cost/budget cost. | Performing Regression Testing increases the product cost/budget cost. |
| 9 | Complete test cases are not executed in the product during this Sanity Testing. | Complete test cases are executed in the product during this Regression Testing. |

## \*\*\*Blocker Bug:-1. In this case, Server crashes and deadlocks for

## major. Configurations, users can't work properly and the server deadlocks when more than 500 users log in.

**\*\*\*\*Definition of Quality Control:-**

Quality control is a product-oriented approach of running a program to determine if it has any defects, as well as making sure that the software meets all of the requirements put forth by the stakeholders.

**\*\*\*\*\*\*\*\*Documentation :-**

Documentation plays a critical role in achieving effective [software testing](https://www.edureka.co/testing-with-selenium-webdriver). Details like requirement specifications, designs, business rules, inspection reports, configurations, code changes, test plans, test cases, bug reports, user manuals, etc. should all be documented.

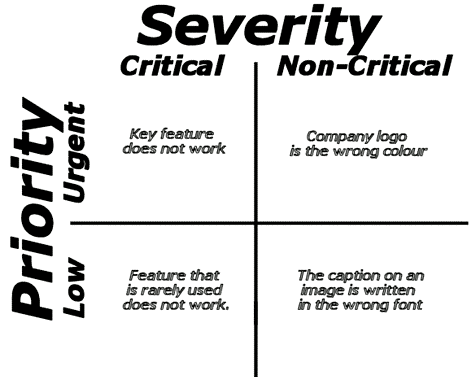
Documenting the test cases will facilitate you to estimate the testing effort you will need along with test coverage and tracking and tracing requirement. Some commonly applied documentation artifacts associated with software testing are:

1. **Test Plan**
2. **Test Scenario**
3. **Test Case**
4. **Traceability Matrix**
5. **Defect Report**

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## \*\*\*Severity and Priority Real-time Examples:-

The priority and severity are combined in four different ways to determine which defect needs immediate attention and which one the least.  Let’s look at some real-time examples to make this concept even clearer.



* **A very high severity with a low priority:**Likewise, for flight operating website, a defect in reservation functionality may be of high severity but can be a low priority as it can be scheduled to release in a next cycle.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |  |  |
| --- | --- | --- | --- |
| **High Severity & High Priority** | **High Severity & Low Priority** | **Low Severity & High Priority** | **Low Severity & Low Priority** |
| You log in to your [amazon.com](https://www.amazon.com/) account, add items to the cart and click the “Proceed to Checkout” button. You make the payment and the system crashes. This defect makes the whole buying functionality unusable and so the severity is high. | Suppose the tester clicks on the “Privacy Notice” hyperlink at the bottom of the amazon.com home page and the page is not displayed. This defect will be of high severity because the functionality is not working. The priority is low because people do not normally spend time reading the privacy notice. | Suppose, that in the amazon.com website, the logo is displayed as ”amazn.com” with the letter “o” missing. This defect does not affect the buying/selling or any other functionality in any way.  So, the severity of this defect is low. But, a mistake in the company logo affects the brand identity and impacts the user experience. So, the defect is of high priority. | Suppose the tester clicks on the “Conditions of Use” hyperlink at the bottom of the amazon.com homepage. If there is an alignment issue in the text displayed or if there is a spelling mistake in the content displayed, the defect is said to be of low priority because people rarely read this page and it does not impact the user experience. The severity is also low because the functionality of the application is not affected. |

### ****High Priority and High Severity Examples****

1. The products added to the cart of an e-commerce website are not visible on the payment page.
2. The login button of the application is not working.

### ****High Priority and Low Severity Examples****

1. The logo of the company’s welcome page is distorted(misrepresented).
2. The action buttons are not visually appealing or the information on the page appears hazy(not clear).

### ****High Severity and Low Priority Examples****

1. If the application is crashing on passing very large input for processing (which is very rarely done).
2. There are some buttons on the website which are overlapping. Although clickable, are creating a fuss(unnecessary interest).

### ****Low Priority and Low Severity Examples****

1. A spelling mistake on the page of the site which is not frequently visited.
2. The color of any text does not match the theme of the website.

**\*\*CI(Continuous Integration):-**

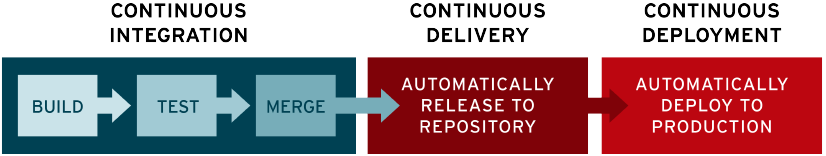
\*\*Successful CI means new code changes to an app are regularly built, tested, and merged to a shared repository. It’s a solution to the problem of having too many branches of an app in development at once that might conflict(Conflict means Opposition)with each other.

\*\*CI/CD:- Both are about automating further stages of the pipeline, but they’re sometimes used separately to illustrate just how much automation is happening.

**\*\*CD(Continuous Delivery and/or Continuous Deployment):-**

Continuous delivery usually means a developer’s changes to an application are automatically bug tested and uploaded to a repository (like GitHub or a container registry), where they can then be deployed to a live production environment by the operations team. It’s an answer to the problem of poor visibility and communication between dev and business teams. To that end, the purpose of continuous delivery is to ensure that it takes minimal effort to deploy new code.

Continuous deployment (the other possible "CD") can refer to automatically releasing a developer’s changes from the repository to production, where it is usable by customers. It addresses the problem of overloading operations teams with manual processes that slow down app delivery. It builds on the benefits of continuous delivery by automating the next stage in the pipeline.



**\*\*\*\*About Agile Model:-**

**In agile methodology, generally 3 meetings are carried out:-**

**1. Daily stand up meeting.**

**2. Sprint meeting.**

**3. Retrospective meeting.**

**Q. Which types of testing are important for web testing?**

**🡪There are 2 types of testing which are very important for Web testing…**

**1)Performance Testing:- Performance Testing** is a testing technique in which quality attributes of a system are measured such as responsiveness, speed under different load conditions and scalability. The performance testing describes which attributes need to be improved before the product is released in the market.

**2)Security Testing:-**Security Testing is a testing technique which determines that the data and resources be saved from the intruders.

**Q. What is the purpose of Entry Criteria?**

**🡪** In case of software testing, entry criteria **defines the conditions to be satisfied in order for the testing to begin** and exit criteria define the conditions that have to be satisfied in order to stop the testing.

### Q. What is Entry and Exit Criteria in STLC?

**🡪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).

**Q. What is the purpose of Exit Criteria?**

**🡪**The exit criteria are used to define the completion of the test level.

**Q. What is the benefit of test independence?**

🡪Test Independence is very useful, because it avoids author bias in defining effective tests.

**Q. What is the difference between Test Scenario & Test Cases?**

**Ans :-In Test Scenario, Only Functionality (Or Scenario) can define. In Test Cases the Functionality (from Scenario) can describe in detail Step By Step.**

### Q. What Test Plans consists of?

🡪 Test design, scope, test strategies, approach are various details that Test plan document consists of.

1. **Test case identifier**
2. **Scope**
3. **Features to be tested**
4. **Features not to be tested**
5. **Test strategy & Test approach**
6. **Test deliverables**
7. **Responsibilities**
8. **Staffing and training**
9. **Risk and Contingencies**

### Q. What are the valuable steps to resolve issues while testing?

**🡪 1.Record : Log and handle any problems which have happened.**

**2.Report : Report the issues to higher level manager.**

**3.Control: Define the issue management process.**

### Q. What all things you should consider before selecting automation tools for the AUT?

🡪**1.Technical Feasibility**

**2.Complexity level**

**3.Application stability**

**4.Test data**

**5.Application size**

**6.Re-usability of automated scripts**

**7.Execution across environment**

### Q. How will you conduct Risk Analysis?

🡪For the risk analysis following steps need to be implemented

1. Finding the score of the risk
2. Making a profile for the risk
3. Changing the risk properties
4. Deploy the resources of that test risk
5. Making a database of risk

### Q. What are the categories of debugging?

🡪 Categories for debugging

1. Brute force debugging
2. Backtracking
3. Cause elimination
4. Program Slicing
5. Fault tree analysis

### Q. Explain how you would allocate a task to team members?

🡪

|  |  |  |
| --- | --- | --- |
| Sr.No | Task | Team Members |
| 1. | Analyze software requirement specification | All the members |
| 2. | Create the test specification | Tester/Test Analyst |
| 3. | Build up the test environment | Test administrator |
| 4. | Execute the test cases | Tester, a Test administrator |
| 5. | Report defects | Tester |

### Q. What does a typical test report contain? What are the benefits of test reports?

🡪A test report contains the following things:

**1.Project Information**

**2.Test Objective**

**3.Test Summary**

**4.Defect**

The benefits of test reports are:

**1.Current status of project and quality of product are informed**

**2.If required, stakeholder and customer can take corrective action**

**3.A final document helps to decide whether the product is ready for release**

### Q) What is the step you would follow once you find the defect?

🡪 Once a defect is found you would follow the steps:-

**a)** **Recreate the defect**

**b) Attach the screenshot**

**c) Log the defect**

### Q). How will you handle a conflict amongst your team members?

1.I will talk individually to each person and note their concerns.

2.I will find a solution to the common problems raised by team members.

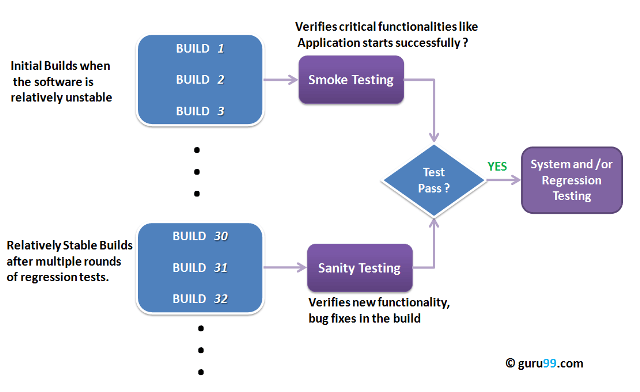
3.I will hold a team meeting, reveal the solution and ask people to co-operate.

|  |  |
| --- | --- |
| StringBuffer Class | StringBuilder Class |
| 1. StringBuffer is present in Java. | 1.StringBuilder was introduced in Java 5. |
| 2. StringBuffer is synchronized. This means that multiple threads cannot call the methods of StringBuffer simultaneously. | 2. StringBuilder is asynchronized. This means that multiple threads can call the methods of StringBuilder simultaneously. |
| 3. Due to synchronization, StringBuffer is called a thread safe class. | 3. Due to its asynchronous nature, StringBuilder is not a thread safe class. |
| 4. Due to synchronization, StringBuffer is lot slower than StringBuilder. | 4. Since there is no preliminary check for multiple threads, StringBuilder is a lot faster than StringBuffer. |

## \*\*\*Blocker Bug:-1. In this case, Server crashes and deadlocks for major configurations, users can't work properly and the server deadlocks when more than 500 users log in.

**\*\*\*\*\*Smoke & Sanity Testing:-**

\*\*\*\*The key differences between Smoke Testing and Sanity Testing can be learned with the help of the following diagram –

  
Smoke Testing vs Sanity Testing

**What is a Software Build?**

If you are developing a simple computer program which consists of only one source code file, you merely need to compile and link this one file, to produce an executable file. This process is very simple.  
Usually, this is not the case. A typical Software Project consists of hundreds or even thousands of source code files. Creating an executable program from these source files is a complicated and time-consuming task.  
You need to use “build” software to create an executable program and the process is called ” *Software Build*”.

## 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.

In [Smoke Testing](https://www.guru99.com/smoke-testing.html), the test cases chose to cover the most important functionality or component of the system. The objective is not to perform exhaustive testing, but to verify that the critical functionalities of the system are working fine.  
For Example, a typical smoke test would be – Verify that the application launches successfully, Check that the GUI is responsive … etc.

## KEY DIFFERENCE

* Smoke Testing has a goal to verify “stability” whereas Sanity Testing has a goal to verify “rationality”( or containing quantities which are expressible).
* Smoke Testing is done by both developers and 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.

## 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.

The objective is “not” to verify thoroughly the new functionality but to determine that the developer has applied some rationality (sanity) while producing the software. For instance, if your scientific calculator gives the result of 2 + 2 =5! Then, there is no point testing the advanced functionalities like sin 30 + cos 50.

## Smoke Testing Vs Sanity Testing – Key Differences

|  |  |  |
| --- | --- | --- |
| **Sr.No** | **Smoke Testing** | **Sanity Testing** |
| 1 | Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine. | Sanity Testing is done to check the new functionality/bugs have been fixed. |
| 2 | The objective of this testing is to verify the “stability” of the system in order to proceed with more rigorous testing. | The objective of the testing is to verify the “rationality” of the system in order to proceed with more rigorous testing. |
| 3 | This testing is performed by the developers or testers. | Sanity testing in software testing is usually performed by testers. |
| 4 | Smoke testing is usually documented or scripted. | Sanity testing is usually not documented and is unscripted. |
| 5 | Smoke testing is a subset of Acceptance testing | Sanity testing is a subset of [Regression Testing](https://www.guru99.com/regression-testing.html). |
| 6 | Smoke testing exercises the entire system from end to end. | Sanity testing exercises only the particular component of the entire system. |
| 7 | Smoke testing is like General Health Check Up. | Sanity Testing is like Specialized Health Check Up. |

## Points to note.

* Both Sanity and Smoke testing are ways to avoid wasting time and effort by quickly determining whether an application is too flawed to merit any rigorous testing.
* Smoke Testing is also called tester [acceptance testing](https://www.guru99.com/user-acceptance-testing.html).
* Smoke testing performed on a particular build is also known as a build verification test.
* One of the best industry practice is to conduct a Daily build and smoke test in software projects.
* Both smoke and sanity tests can be executed manually or using an [automation tool](https://www.guru99.com/automated-testing-tools.html).  When automated tools are used, the tests are often initiated by the same process that generates the build itself.
* As per the needs of testing, you may have to execute both Sanity and Smoke Tests in the software build. In such cases, you will first execute Smoke tests and then go ahead with Sanity Testing. In industry, test cases for Sanity Testing are commonly combined with that for smoke tests, to speed up test execution. Hence, it’s a common that the terms are often confused and used interchangeably.