**Module – 2 Manual Testing**

* **What is Exploratory Testing?**

Exploratory testing is an approach to [software testing](https://www.atlassian.com/continuous-delivery/software-testing) that is often described as simultaneous learning, test design, and execution. It focuses on discovery and relies on the guidance of the individual tester to uncover defects that are not easily covered in the scope of other tests.

* **What is traceability matrix?**

A traceability matrix is a document that details [the technical requirements](https://www.wrike.com/project-management-guide/faq/what-are-technical-requirements-in-project-management/) for a given [test scenario](https://learn.marsdd.com/article/product-testing-methods/#:~:text=Test%20cases%20probe%20each%20way,changes%20that%20might%20be%20necessitated.) and its current state. It helps the testing team understand the level of testing that is done for a given product. The traceability process itself is used to review the test cases that were defined for any requirement. It helps users identify which [requirements](https://www.wrike.com/blog/requirements-gathering-guide/#What-is-requirements-gathering-in-project-management) produced the most number of defects during a testing cycle.

* **What is Boundary value testing?**

[Boundary Value Analysis](https://www.geeksforgeeks.org/boundary-value-analysis-triangle-problem/) is based on testing the boundary values of valid and invalid partitions. The behaviour at the edge of the equivalence partition is more likely to be incorrect than the behaviour within the partition, so boundaries are an area where testing is likely to yield defects. It checks for the input values near the boundary that have a higher chance of error. Every partition has its maximum and minimum values and these maximum and minimum values are the boundary values of a partition.

* **What is Equivalence partitioning testing?**

Equivalence Partitioning Method is also known as Equivalence class partitioning (ECP). It is a [software testing](https://www.geeksforgeeks.org/software-testing-basics/) technique or [black-box testing](https://www.geeksforgeeks.org/software-engineering-black-box-testing/) that divides input domain into classes of data, and with the help of these classes of data, test cases can be derived. An ideal test case identifies class of error that might require many arbitrary test cases to be executed before general error is observed.

* **What is Integration testing?**

Integration testing -- also known as integration and testing (I&T) -- is a type of [software testing](https://www.techtarget.com/whatis/definition/software-testing) in which the different units, modules or components of a software application are tested as a combined entity. However, these modules may be coded by different programmers.

* **What determines the level of risk?**

The likelihood of an adverse event and the impact of the event.

* **What is Alpha testing?**

Alpha testing is the initial phase of validating whether a new product will perform as expected. Alpha tests are carried out early in the development process by internal staff and are followed up with beta tests, in which a sampling of the intended audience actually tries the product out.

* **What is beta testing?**

In software development, a beta test is the second phase of software testing in which a sampling of the intended audience tries the product out. Beta is the second letter of the Greek alphabet. Originally, the term alpha test meant the first phase of testing in a software development process.

* **What is component testing?**

Component testing, also known as program or module testing, is done after unit testing. In this type of testing those test objects can be tested independently as a component without integrating with other components e.g., modules, classes, objects, and programs. This testing is done by the development team.

* **What is functional system testing?**

Functional testing is a type of testing that seeks to establish whether each application feature works as per the software requirements. Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user’s expectations. The testing is done by providing sample inputs, capturing resulting outputs, and verifying that actual outputs are the same as expected outputs.

* **What is Non-Functional Testing?**

Non-functional testing assesses application properties that aren't critical to functionality but contribute to the end-user experience. Performance and reliability under load aren't functional components of a software system but can certainly make or break the user experience.

* **What is GUI Testing?**

**GUI Testing** is a software testing type that checks the Graphical User Interface of the Software. The purpose of Graphical User Interface (GUI) Testing is to ensure the functionalities of software application work as per specifications by checking screens and controls like menus, buttons, icons, etc.

* **What is Adhoc testing?**

Performing random testing without any plan is known as Ad Hoc Testing.  It is also referred to as Random Testing or Monkey Testing. This type of testing doesn’t follow any documentation or plan to perform this activity. The testing steps and the scenarios only depend upon the tester, and defects are found by random checking.

* **What is load testing?**

Load testing is the process of putting simulated demand on software, an application or website in a way that tests or demonstrates its behaviour under various conditions. Load testing is about creating production simulations within an application or system that is as near as possible to being a finished product ready to deploy and subject to the masses.

* **What is stress Testing?**

Stress Testing is a software testing technique that determines the robustness of software by testing beyond the limits of normal operation. Stress testing is particularly important for critical software but is used for all types of software. Stress testing emphasizes robustness, availability, and error handling under a heavy load rather than what is correct behaviour under normal situations. Stress testing is defined as a type of software testing that verifies the stability and reliability of the system. This test particularly determines the system on its robustness and error handling under extremely heavy load conditions. It even tests beyond the normal operating point and analyses how the system works under extreme conditions. Stress testing is performed to ensure that the system would not crash under crunch situations. Stress testing is also known as Endurance Testing or Torture Testing.

* **What is white box testing and list the types of white box testing?**

**White Box Testing** is a testing technique in which software’s internal structure, design, and coding are tested to verify input-output flow and improve design, usability, and security. In white box testing, code is visible to testers, so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing.

Following is important White Box Testing Techniques:

* Statement Coverage
* Decision Coverage
* Branch Coverage
* Condition Coverage
* Multiple Condition Coverage
* Finite State Machine Coverage
* Path Coverage
* Control flow testing
* Data flow testing

**What is black box testing? What are the different black box testing techniques?**

Black box testing involves testing a system with no prior knowledge of its internal workings. A tester provides an input, and observes the output generated by the system under test. This makes it possible to identify how the system responds to expected and unexpected user actions, its response time, usability issues and reliability issues. Black box testing is a powerful testing technique because it exercises a system end-to-end. Just like end-users “don’t care” how a system is coded or architected, and expect to receive an appropriate response to their requests, a tester can simulate user activity and see if the system delivers on its promises. Along the way, a black box test evaluates all relevant subsystems, including UI/UX, web server or application server, database, dependencies, and integrated systems.

Following is the type of black box Testing:

1. **Equivalence Partitioning**
2. **Boundary Value Analysis**
3. **Decision Table Testing**
4. **State Transition Testing**
5. **Error Guessing**

* **Mention what are the categories of defects?**

A software bug arises when the expected result doesn’t match with the actual results. It can also be error, flaw, failure, or fault in a computer program. Most bugs arise from mistakes and errors made by developers, architects.

Following are the methods for preventing programmers from introducing bugs during development:

* **Programming Techniques adopted**
* **Software Development methodologies**
* **Peer Review**
* **Code Analysis**
* **Arithmetic Defects**
* **Logical Defects**
* **Syntax Defects**
* **Multithreading Defects**
* **Interface Defects**
* **Performance Defects**
* **Mention what big bang testing is?**

Big Bang Integration Testing is an integration testing strategy wherein all units are linked at once, resulting in a complete system. When this type of testing strategy is adopted, it is difficult to isolate any errors found, because attention is not paid to verifying the interfaces across individual units.

* **What is the purpose of exit criteria?**

Exit criterion is used to determine whether a given test activity has been completed or NOT. Exit criteria can be defined for all of the test activities right from planning, specification and execution. Exit criterion should be part of test plan and decided in the planning stage.

* **When should "Regression Testing" be performed?**

Regression testing is done after functional testing has concluded, to verify that the other functionalities are working. In the corporate world, regression testing has traditionally been performed by a software quality assurance team after the development team has completed work.

* **What is 7 key principles? Explain in detail?**

7 Principles of Software Testing

1. **Testing shows presence of defects**
2. **Exhaustive testing is not possible**
3. **Early testing**
4. **Defect clustering**
5. **Pesticide paradox**
6. **Testing is context dependent**
7. **Absence of errors fallacy**

**1.Testing shows a presence of defects**

testing principle states that – Testing talks about the presence of defects and don’t talk about the absence of defects. i.e., Software Testing reduces the probability of undiscovered defects remaining in the software but even if no defects are found, it is not a proof of correctness.

But what if, you work extra hard, taking all precautions & make your software product 99% bug-free. And the software does not meet the needs & requirements of the clients.

This leads us to our next principle, which states that- Absence of Error

**2.Exhaustive testing is not possible**

Yes! Exhaustive testing is not possible. Instead, we need the optimal amount of testing based on the risk assessment of the application.

And the million-dollar question is, how do you determine this risk?

To answer this let’s do an exercise

In your opinion, which operation is most likely to cause your Operating system to fail?

I am sure most of you would have guessed, Opening 10 different application all at the same time.

So, if you were testing this Operating system, you would realize that defects are likely to be found in multi-tasking activity and need to be tested thoroughly which brings us to our next principle[Defect](https://www.guru99.com/defect-management-process.html)Clustering.

**3.Defect Clustering**

Defect Clustering which states that a small number of modules contain most of the defects detected. This is the application of the Pareto Principle to software testing: approximately 80% of the problems are found in 20% of the modules.

By experience, you can identify such risky modules. But this approach has its own problems

If the same tests are repeated over and over again, eventually the same test cases will no longer find new bugs.

**4.Pesticide Paradox**

Repetitive use of the same pesticide mixes to eradicate insects during farming will over time lead to the insect’s developing resistance to the pesticide Thereby ineffective of pesticides on insects. The same applies to software testing. If the same set of repetitive tests are conducted, the method will be useless for discovering new defects.

To overcome this, the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects.

Testers cannot simply depend on existing test techniques. He must look out continually to improve the existing methods to make testing more effective. But even after all this sweat & hard work in testing, you can never claim your product is bug-free. To drive home this point, let’s see this video of the public launch of Windows 98

You think a company like MICROSOFT would not have tested their OS thoroughly & would risk their reputation just to see their OS crashing during its public launch!

**5.Absence of Error – fallacy**

It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement. Software testing is not mere finding defects, but also to check that software addresses the business needs. The absence of Error is a Fallacy i.e., Finding and fixing defects does not help if the system build is unusable and does not fulfil the user’s needs & requirements.

To solve this problem, the next principle of testing states that Early Testing.

**6.Early Testing**

Early Testing – Testing should start as early as possible in the Software Development Life Cycle. So that any defects in the requirements or design phase are captured in early stages. It is much cheaper to fix a Defect in the early stages of testing. But how early one should start testing? It is recommended that you start finding the bug the moment the requirements are defined. More on this principle in a later training tutorial.

**7.Testing is context dependent**

Testing is context dependent which basically means that the way you test an e-commerce site will be different from the way you test a commercial off the shelf application. All the developed software’s are not identical. You might use a different approach, methodologies, techniques, and types of testing depending upon the application type. For instance, testing, any POS system at a retail store will be different than testing an ATM machine.

* **Difference between QA v/s QC v/s Tester**

To summarize, Quality assurance is a set of processes that help “avoid” defects and assure quality. While Quality Control is a set of activities that help detect defects and quality issues before the products reach the hands of end customers. Testing is one of the ways of detecting those defects.

* **Difference between Smoke and Sanity?**

Smoke Testing is performed to ascertain that the critical functionalities of the program are working fine.

Sanity testing is done at random to verify that each functionality is working as expected.

* **Difference between verification and Validation**

Verification is a process of determining if the software is designed and developed as per the specified requirements.

Validation is the process of checking if the software (end product) has met the client's true needs and expectations.

* **Explain types of Performance testing.**

The different types of performance testing:

1. **Load Testing**
2. **Stress Testing**
3. **Scalability Testing**
4. **Spike Testing**
5. **Soak Testing**
6. **Configuration Testing**

* **What is Error, Defect, Bug and failure?**

We can say that a mistake made by a programmer during coding is called an **error**,

an error found during the unit testing in the development phase is called a **defect**,

an error found during the testing phase is called a **bug**.

when an error is found at an end user’s end is called as the **failure.**

* **Difference between Priority and Severity**

Severity is basically a parameter that denotes the total impact of a given defect on any software. Severity relates to the standards of quality.

Priority is basically a parameter that decides the order in which we should fix the defects. Priority relates to the scheduling of defects to resolve them in software.

* **What is Bug Life Cycle?**

Defect life cycle, also known as Bug Life cycle is the journey of a defect cycle, which a defect goes through during its lifetime. It varies from organization to organization and also from project to project as it is governed by the software testing process and also depends upon the tools used.

**Explain the difference between Functional testing and Non-Functional testing**

| Functional Testing | Non-functional Testing |
| --- | --- |
| It verifies the operations and actions of an application. | It verifies the behaviour of an application. |
| It is based on requirements of customer. | It is based on expectations of customer. |
| It helps to enhance the behaviour of the application. | It helps to improve the performance of the application. |
| Functional testing is easy to execute manually. | It is hard to execute non-functional testing manually. |
| It tests what the product does. | It describes how the product does. |
| Functional testing is based on the business requirement. | Non-functional testing is based on the performance requirement. |

* **What is the difference between test scenarios, test cases, and test script?**

A test scenario is any functionality that a software testing company can examine. It is also called a Test Condition or Test Possibility.

A test case is a document that lists the steps a QA engineer needs to execute.

A test script is a short program written in a programming language.

* **Explain what Test Plan is? What is the information that should be covered?**

A Test Plan refers to a detailed document that catalogues 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.

Components of a Test Plan

1. **Scope:** Details the objectives of the particular project. Also, it details user scenarios to be used in tests. If necessary, the scope can specify what scenarios or issues the project will not cover.
2. **Schedule:** Details start dates and deadlines for testers to deliver results.
3. **Resource Allocation:** Details which tester will work on which test.
4. **Environment:** Details the nature, configuration, and availability of the test environment.
5. **Tools:** Details what tools are to be used for testing, bug reporting, and other relevant activities.
6. **Defect Management:** Details how bugs will be reported, to whom and what each bug report needs to be accompanied by. For example, should bugs be reported with screenshots, text logs, or videos of their occurrence in the code?
7. **Risk Management:** Details what risks may occur during software testing, and what risks the software itself may suffer if released without sufficient testing.
8. **Exit Parameters:** Details when testing activities must stop. This part describes the results that are expected from the QA operations, giving testers a benchmark to compare actual results to.

* **What is priority?**

Priority is defined as the order in which the defects should be resolved. The priority status is usually set by the testing team while raising the defect against the dev team mentioning the timeframe to fix the defect. The Priority status is set based on end user’s requirement.

* **What is severity?**

Severity is defined as the extent to which a particular defect can create an impact on the software. Severity is a parameter to denote the implication and the impact of the defect on the functionality of the software.

* **What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?**

|  |  |
| --- | --- |
| **SDLC** | **STLC** |
| SDLC, or software development life cycle, relates mainly to software development and includes all phases of software development, including testing. | In essence, STLC is related to software testing, meaning that it is a software testing process that entails several phases. |
| As a whole, it covers the entire life cycle of the software and can be considered the predecessor. | Since it is part of SDLC and only involves testing, it is considered a child or successor. |
| SDLC aims to manage the entire process of software development from start to finish and to deliver a quality product that meets customer needs. | The focus is solely on test development and helps to make the testing process more sophisticated, consistent, and useful. |
| Phases of the SDLC are completed before those of the STLC. | Phases of STLC are carried out after the phases of SDLC. |
| Business Analysts and Product Analysts collect requirements and prepare a Development Plan during the Requirements collection phase of the SDLC. | The QA (Quality Assurance) team will analyse requirement documents such as functional and non-functional requirements. |
| The SDLC includes the following phases: Requirements Collection/Planning Analysis/Defining Designing the software programming or Coding (Building the Software) Testing Deployment/Installation Maintenance | The STLC includes the following phases: Requirement Analysis Test Planning Test Development Test Environment Setup Test Execution Test Closure |
| Throughout the SDLC process, the intent is to overcome any hurdle on the way to successful software development. | Testing is only intended to find any weaknesses or pitfalls in the system. |
| SDLC involves planning and designing software based on the requirements of the development team. | STLC involves the planning of tests by the testing team (Test Architect or Test Lead). |
| In this phase, programmers start writing code according to the designed document in any programming language to build the system from scratch. | Test cases and test scripts are developed by the testing team (quality assurance team) to verify the product’s quality. They prepare the test environment and execute the tests. |
| As soon as the development team has written the code, they set up a test environment to validate it. | The testers ensure the test environment is prepared based on the prerequisites and conduct smoke tests to determine if the environment is stable enough for testing the product. |
| The objective of this phase is to test the software. Among the activities included in this testing are Unit, Integration, System, Retest & Regression testing, etc., and the development team also participates in fixing reported bugs. | System integration testing is then conducted based on the test cases. All bugs and errors are reported, retested, and fixed. The product is also subject to regression tests and is signed off as soon as it meets the exit criteria. |
| As soon as the application has been approved by the various testing teams, it is deployed in a production environment for real end-users. | After the product has been deployed, smoke testing and sanity testing take place in the production environment, and the testing team prepares test reports and an analysis matrix for analysing the product. |
| If necessary, post-deployment support, enhancement, and update are included. | The QA team runs regression tests to check deployed maintenance code. The team maintains test cases and automated scripts to make sure that tests are updated. |
| Throughout the SDLC, more people (developers) are needed. | The QA team runs regression tests to check deployed maintenance code. The team maintains test cases and automated scripts to make sure that tests are updated. |
| The end result of SDLC is the creation of reusable software systems. | STLC results in a tested software system. |

* **Advantage of Bugzilla.**

**The Advantages of Bugzilla are:**

1. it is an open-source widely used bug tracker;
2. it is easy in usage and its user interface is understandable for people without technical knowledge;
3. it easily integrates with test management instruments;
4. it integrates with an e-mailing system;
5. it automates documentation.

* **Difference between priority and severity**

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Severity in Testing** | **Priority in Testing** |
| Definition | Severity is a term that denotes how severely a defect can affect the functionality of the software. | Priority is a term that defines how fast we need to fix a defect. |
| Parameter | Severity is basically a parameter that denotes the total impact of a given defect on any software. | Priority is basically a parameter that decides the order in which we should fix the defects. |
| Relation | Severity relates to the standards of quality. | Priority relates to the scheduling of defects to resolve them in software. |
| Value | The value of severity is objective. | The value of priority is subjective. |
| Change of Value | The value of Severity changes continually from time to time. | The value of Priority changes from time to time. |
| Who Decides the Defect | The testing engineer basically decides a defect’s severity level. | The product manager basically decides a defect’s priority level. |
| Types | There are 5 types of Severities: Cosmetic, Minor, Moderate, Major, and Critical. | There are 3 types of Priorities: High, Medium, and Low. |

* **What are the different Methodologies in Agile Development Model?**

1. **Kanban**

Kanban is a simple, visual means of managing projects that enables teams to see the progress so far and what’s coming up next. Kanban projects are primarily managed through a Kanban board, which segments tasks into three columns: “To Do,” “Doing,” and “Done.”

1. **Scrum**

Scrum is similar to Kanban in many ways. Scrum typically uses a Scrum board, similar to a Kanban board, and groups tasks into columns based on progress. Unlike Kanban, Scrum focuses on breaking a project down into sprints and only planning and managing one sprint at a time. Scrum also has unique project roles: Scrum master and product owner.

1. **Extreme Programming (XP)**

Extreme Programming (XP) was designed for Agile software development projects. It focuses on continuous development and customer delivery and uses intervals or sprints, similar to a Scrum methodology. However, XP also has 12 supporting processes specific to the world of software development:

1. **Feature-driven development (FDD)**

Feature-driven development is another software-specific Agile framework. This methodology involves creating software models every two weeks and requires a development and design plan for every model feature. It has more rigorous documentation requirements than XP, so it’s better for teams with advanced design and planning abilities. FDD breaks projects down into five basic activities:

1. **Dynamic Systems Development Method (DSDM)**

The Dynamic Systems Development Method (DSDM) was born of the need for a common industry framework for rapid software delivery. Rework is to be expected, and any development changes that occur must be reversible. Like Scrum, XP, and FDD, DSDM uses sprints. This framework is based on eight fundamental principles:

1. **Crystal**

Crystal is a family of Agile methodologies that includes Crystal Clear, Crystal Yellow, Crystal Orange, Crystal Red, etc. Each has a unique framework. Your choice depends on several project factors, such as your team size, priorities, and project criticality.

1. **Lean**

Lean development is often grouped with Agile, but it’s an entirely different methodology that happens to share many of the same values.

* **Explain the difference between Authorization and Authentication in Web testing. What are the common problems faced in Web testing?**

authentication is the process of verifying who someone is, whereas authorization is the process of verifying what specific applications, files, and data a user has access to. The situation is like that of an airline that needs to determine which people can come on board.

* **When to used Usability Testing?**

If possible, usability testing can and should be conducted on the current iteration of a product before beginning any new design work, after you’ve begun the strategy work around a brand-new site or app.

* **What is the procedure for GUI Testing?**

It extensively checks the user-interface of the application under test.

* Testing the size, position, height, width of the visual elements
* Verifying and testing the error messages are displayed or not
* Testing different sections of the display screen
* Verifying the usability of carousel arrows
* Checking the navigation elements at the top of the page
* Checking the message displayed, frequency and content
* Verifying the functionality of proper filters and ability to retrieve results.
* Checking alignment of radio buttons, drop downs
* Verifying the title of each section and their correctness
* Cross-checking the colours and its synchronization with the theme.