*Software Testing Assignment*

Module – 2 (Manual Testing)

1. What is Exploratory Testing?

Exploratory Testing is a type of software testing in which the tester is free to select any possible methodology to test the software. It is an unscripted approach to software testing. In exploratory testing, software developers use their learning, knowledge, skills, and abilities to test the software developed by themselves. Exploratory testing checks the functionality and operations of the software as well as identify the functional and technical faults in it. Exploratory testing aims to optimize and improve the software in every possible way. The exploratory testing technique combines the experience of testers with a structured approach to testing. It is often performed as a black box testing technique.

1. What is traceability matrix?

A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship. It is used to track the requirements and to check the current project requirements are met.

1. What is Boundary value testing?

Boundary Value Analysis 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.

* A boundary value for a valid partition is a valid boundary value.
* A boundary value for an invalid partition is an invalid boundary value.
* For each variable we check-
* Minimum value.
* Just above the minimum.
* Nominal Value.
* Just below Max value.
* Max value.

1. What is Equivalence partitioning testing?

Equivalence Partitioning Method is also known as Equivalence class partitioning (ECP). It is a software testing technique or 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.

In equivalence partitioning, equivalence classes are evaluated for given input conditions. Whenever any input is given, then type of input condition is checked, then for this input conditions, Equivalence class represents or describes set of valid or invalid states.

1. What is Integration testing?

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

1. What determines the level of risk?

Risk management is the process of identifying, analyzing, evaluating, and addressing potential risks in an organization.

Risk management involves the following four stages:

* Identification: Identify potential risks that could negatively impact the organization’s objectives and goals.
* Analysis: Analyze the identified risks to determine their likelihood and potential impact.
* Evaluation: Evaluate the risks based on their likelihood and potential impact to determine which one’s merit attention.
* Treatment: Address the identified risks using appropriate strategies and techniques to mitigate or eliminate them.

1. What is Alpha testing?

Alpha Testing is a type of software testing performed to identify bugs before releasing the product to real users or to the public. Alpha Testing is one of the user acceptance testing. This is referred to as alpha testing only because it is done early on, near the end of the development of the software. Alpha testing is commonly performed by homestead software engineers or quality assurance staff. It is the last testing stage before the software is released into the real world.

Phases of Alpha Testing

* Planning
* Preparation
* Execution
* Evaluation
* Reporting
* Closure

1. What is beta testing?

Beta testing is the process of testing a software product or service in a real-world environment before its official release. It is an essential step in the software development lifecycle as it helps identify bugs and errors that may have been missed during the development process.

Types of Beta Testing

* Traditional Beta testing
* Public Beta Testing
* Technical Beta Testing
* Focused Beta Testing
* Post-release Beta Testing

1. What is component testing?

Component testing is defined as a software testing type, in which the testing is performed on each individual component separately without integrating with other components. It’s also referred to as Module Testing when it is viewed from an architecture perspective. Component Testing is also referred to as Unit Testing, Program Testing or Module Testing.

1. What is functional system testing?

Functional Testing is a type of software testing that validates the software system against the functional requirements/specifications. The purpose of Functional tests is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements.

Functional testing mainly involves black box testing and it is not concerned about the source code of the application. This testing checks User Interface, APIs, Database, Security, Client/Server communication and other functionality of the Application Under Test. The testing can be done either manually or using automation.

Functional Testing Types

* Unit testing
* Smoke testing
* User Acceptance
* Integration Testing
* Regression testing
* Localization
* Globalization
* Interoperability

1. What is Non-Functional Testing?

Non-Functional Testing is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc) 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.

Types of Non-Functional Testing

* Performance Testing
* Load Testing
* Compatibility Testing
* Usability Testing
* Stress Testing
* Maintainability Testing
* Scalability Testing
* Security Testing

1. What is GUI Testing?

Graphical User Interface Testing (GUI) Testing is the process for ensuring proper functionality of the graphical user interface (GUI) for a specific application. GUI testing generally evaluates a design of elements such as layout, colors and also fonts, font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links, and content. GUI testing processes may be either manual or automatic and are often performed by third-party companies, rather than developers or end users.

1. What is Adhoc testing?

Adhoc testing is a type of software testing which is performed informally and randomly after the formal testing is completed to find out any loophole in the system. For this reason, it is also known as Random testing or Monkey testing. Adhoc approach. That’s why Adhoc testing is a type of Unstructured Software Testing.

Adhoc testing has –

* No Documentation
* No Test cases
* No Test Design

Types of Adhoc Testing:

* Buddy Testing
* Pair Testing
* Monkey Testing

1. What is load testing?

Load testing is a type of Performance Testing that determines the performance of a system, software product, or software application under real-life-based load conditions. Load testing determines the behaviour of the application when multiple users use it at the same time.

During load testing, various scenarios are simulated to test the system’s behavior under different load conditions. This can include simulating a high number of concurrent users, simulating numerous requests, and simulating heavy network traffic. The system’s performance is then measured and analyzed to identify any bottlenecks or issues that may occur.

1. 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 behavior 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.

Types of Stress Testing:

* Transactional Stress Testing
* Systematic Stress Testing
* Server-client Stress Testing
* Product Stress Testing
* Application Stress Testing

1. What is white box testing and list the types of white box testing?

White box testing techniques analyse the internal structures the used data structures, internal design, code structure, and the working of the software rather than just the functionality as in black box testing. It is also called glass box testing or clear box testing or structural testing. White Box Testing is also known as transparent testing or open box testing.

White box testing is a software testing technique that involves testing the internal structure and workings of a software application. The tester has access to the source code and uses this knowledge to design test cases that can verify the correctness of the software at the code level.

Types of White Box Testing

* Unit Testing
* Static Analysis
* Dynamic Analysis
* Statement Coverage
* Branch Testing
* Path Testing
* Loop Testing

1. What is black box testing? What are the different black box testing techniques?

Black-box testing is a type of software testing in which the tester is not concerned with the internal knowledge or implementation details of the software but rather focuses on validating the functionality based on the provided specifications or requirements.

The testing techniques are:

* Equivalent Partitioning
* Boundary Value Analysis
* Decision Tables
* State Transition Testing
* Use Case Testing
* Syntax or Pattern Testing

1. Mention what are the categories of defects?

Software Defect is some kind of error, flaw or some kind of mistake from the development team which prevent the software from the smooth working. It directly affects software quality; software quality is something how smooth and reliable your software is. Smoothness and reliability are how less defects your software have.

Categories of defects:

* Errors of commissions
* Errors of omissions
* Errors of clarity
* Error of speed and capacity

1. 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.

1. What is the purpose of exit criteria?

Exit criteria are the defined requirements within software testing that must be met in order to determine that testing has been completed. These conditions are typically defined by engineering or test leadership to ensure quality standards are met.

Software testing teams will use exit criteria to determine if a test plan or project can exit to the next stage or be considered complete. This isn't something that should be left up to the subjective and/or ad hoc decisions of a test admin or SQA engineer, as it can directly impact the success of the next stage or project as a whole.

1. When should "Regression Testing" be performed?

Regression testing is a type of software testing that ensures that existing functionality works as expected after making changes to the code, such as adding new features, fixing bugs, or improving performance. Regression testing helps to avoid introducing new defects or breaking existing features while modifying the software. In this article, you will learn when you should perform regression testing and what factors to consider when planning and executing it.

It depends on the scope and complexity of the changes, the type and level of testing, the available resources and time, and the testing strategy and methodology.

1. What is 7 key principles? Explain in detail?

* Testing shows presence of defects
* Exhaustive testing is not possible
* Early testing
* Defect clustering
* Pesticide paradox
* Testing is context dependent
* Absence of errors fallacy

1. Testing shows presence of defects:

The goal of software testing is to make the software fail. Software testing reduces the presence of defects. Software testing talks about the presence of defects and doesn’t talk about the absence of defects. Software testing can ensure that defects are present but it cannot prove that software is defect-free. Even multiple testing can never ensure that software is 100% bug-free. Testing can reduce the number of defects but not remove all defects.

1. Exhaustive testing is not possible:

It is the process of testing the functionality of the software in all possible inputs (valid or invalid) and pre-conditions is known as exhaustive testing. Exhaustive testing is impossible means the software can never test at every test case. It can test only some test cases and assume that the software is correct and it will produce the correct output in every test case. If the software will test every test case, then it will take more cost, effort, etc., which is impractical.

1. Early testing:

To find the defect in the software, early test activity shall be started. The defect detected in the early phases of SDLC will be very less expensive. For better performance of software, software testing will start at the initial phase i.e. testing will perform at the requirement analysis phase.

1. Defect clustering:

In a project, a small number of modules can contain most of the defects. The Pareto Principle for software testing states that 80% of software defects come from 20% of modules.

1. Pesticide paradox:

Repeating the same test cases, again and again, will not find new bugs. So it is necessary to review the test cases and add or update test cases to find new bugs.

1. Testing is context dependent:

The testing approach depends on the context of the software developed. Different types of software need to perform different types of testing. For example, the testing of the e-commerce site is different from the testing of the Android application.

1. Absence of errors fallacy:

If a built software is 99% bug-free but does not follow the user requirement then it is unusable. It is not only necessary that software is 99% bug-free but it is also mandatory to fulfil all the customer requirements.

1. Difference between QA v/s QC v/s Tester

The quality of a product, application, or website plays a vital role in its success. While developers build the functionalities, testers’ job is to determine the software’s quality and effectiveness. However, the definition of quality can be subjective, making it crucial to establish methods and processes to verify the product’s quality for both those involved in creating it and its users.

In software development, the terms ‘Testing,’ ‘Quality Assurance,’ and ‘Quality Control’ are often thrown around interchangeably. But it’s important to understand that Quality Control vs Quality Assurance vs Testing are distinct concepts that form different parts of the quality management cycle. In this blog, we delve into the significance of these three terms and explore how they can be utilized to enhance the quality of a product.

This blog discusses the differences between Quality Control, Quality Assurance, and Testing. These are all essential parts of ensuring software works well, so understanding how they work together is vital if you want a great product. We’ll look at these processes and explain what makes them unique. If you’re into software testing or engineering or are just curious about software development, this blog will be super helpful! Let’s start exploring the world of Quality Control vs Quality Assurance vs Testing!

1. Difference between Smoke and Sanity?

* 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.

1. Difference between verification and Validation

Verification is the process of checking that a software achieves its goal without any bugs. It is the process to ensure whether the product that is developed is right or not. It verifies whether the developed product fulfils the requirements that we have. Verification is static testing.

Validation is the process of checking whether the software product is up to the mark or in other words product has high level requirements. It is the process of checking the validation of product i.e. it checks what we are developing is the right product. it is validation of actual and expected product. Validation is the dynamic testing.

1. Explain types of Performance testing

* Load testing:

Load testing is a type of testing which involves evaluating the performance of the system under the expected workload. A typical load test includes determining the response time, throughput, error rate, etc during the course of the load test.

* Stress testing:

Stress testing is a type of performance testing where we evaluate the application’s performance at a load much higher than the expected load. Another aspect of the stress testing is to determine the break-point of the application, the point at which the application fails to respond in the correct manner.

* Endurance testing

Endurance testing is also known as ‘Soak Testing’. It is done to determine if the system can sustain the continuous expected load for a long duration. Issues like memory leakage are found with endurance testing.

* Spike testing

In spike testing, we analyse the behavior of the system on suddenly increasing the number of users. It also involves checking if the application is able to recover after the sudden burst of users.

* Volume testing

The volume testing is performed by feeding the application with a high volume of data. The application can be tested with a large amount of data inserted in the database or by providing a large file to the application for processing. Using volume testing, we can identify the bottleneck in the application with a high volume of data.

1. What is Error, Defect, Bug and failure?

A bug refers to defects which means that the software product or the application is not working as per the adhered requirements set. When we have any type of logical error, it causes our code to break, which results in a bug.

A defect refers to a situation when the application is not working as per the requirement and the actual and expected result of the application or software are not in sync with each other.

Error is a situation that happens when the Development team or the developer fails to understand a requirement definition and hence that misunderstanding gets translated into buggy code. This situation is referred to as an Error and is mainly a term coined by the developers.

Failure is the accumulation of several defects that ultimately lead to Software failure and results in the loss of information in critical modules thereby making the system unresponsive. Generally, such situations happen very rarely because before releasing a product all possible scenarios and test cases for the code are simulated. Failure is detected by end-users once they face a particular issue in the software.

1. Difference between Priority and Severity

Bug Severity or Defect Severity in testing is a degree of impact a bug or a Defect has on the software application under test. A higher effect of bug/defect on system functionality will lead to a higher severity level. A Quality Assurance engineer usually determines the severity level of a bug/defect.

Priority is defined as the order in which a defect should be fixed. Higher the priority the sooner the defect should be resolved. Defects that leave the software system unusable are given higher priority over defects that cause a small functionality of the software to fail.

1. What is Bug Life Cycle?

Bug Life Cycle in software testing is the specific set of states that defect or bug goes through in its entire life. The purpose of Defect life cycle is to easily coordinate and communicate current status of defect which changes to various assignees and make the defect fixing process systematic and efficient.

Bug Status in defect life cycle is the present state from which the defect or a bug is currently undergoing. The goal of defect status is to precisely convey the current state or progress of a defect or bug in order to better track and understand the actual progress of the defect life cycle.

1. Explain the difference between Functional testing and Non-Functional testing