Manual Testing Material

Introduction to Software Testing

What is Software Testing ?

1.1 "Software testing is a process of executing the application with the intent of finding the defects by comparing the output behavior of the application with expected behavior (requirement)." In other words it's comparing the actual behavior of an application with expected behavior.

1.2 Why Software Testing Humans make mistakes all the time!!

"Software testing is really required to point out the defects and errors that were made during the development phases". We humans can't identify our mistakes in a work done by us. We should get someone else to check our work because another person may identify the mistakes done by us. In the same way software developers may not identify the mismatches in a program or application implemented by them which can be identify by the another department called Software Test Engineer.

1.3 Benefits of Software Testing

"Software testing helps in finalizing the software application against business requirements." Software testing makes sure that the testing is being done properly and hence the system is ready for the customers to use. Below are few benefits of software testing.

- Finding the defects before delivery
- Gaines the confidence about quality

- To Prevent defects - Ensure the requirements are delivered to client

1.4 What is Quality

"Software quality is nothing but delivering a bug free application and delivered on time with all requirements."

ISO 8402-1986 standard defines quality as "the totality of features and characteristics of a product or service that bears its ability to satisfy stated or implied needs."

1.5 What is defect

"A defect is a deviation or mismatch from the requirements". When actual result deviates from the expected result while testing a software application or product then it results into a defect. Hence, any deviation from the specification mentioned in the functional specification document is a defect. In different organizations it's called differently like bug, issue, incidents or problem.

1.6 Project Vs Product

"Project is developed for a single customer on his own requirements by the software companies and the project will be used by the customer only." "Product is developed for multiple customers on their consolidated requirements by the software companies and the product will be used by all customers."

2. Software Development Life Cycle (SDLC)

2.1 What is Software Development Life Cycle

“SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software”

2.2 Why Software Development Life Cycle

“SDLC ensure success in process of software development.”

2.3 Phases of Software Development Life Cycle

- Initial
- Analysis
- Design
- Coding
- Testing
- Delivery & Maintenance

2.3.1 **Initial**

“Business requirements are gathered in this phase.

“ This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements like; - Who is going to use the system? - How will they use the system? - What data should be input into the system? - What data should be output by the system? Roles Involved: Business Analyst (BA), System Architects Outcome: System Requirement Specification (SRS)

2.3.2 **Analysis** “After requirement gathering these requirements are analyzed for their validity and the possibility of developing the

requirements in the system." Requirement analysis is the most important and fundamental stage in SDLC. It is performed by both development team and testing team. Roles Involved: Dev & QA team, Architects, Project Managers Outcome: Final SRS approved by customer, Technology selection for both Dev & QA

2.3.3 **Design** "During this part of the design phase, the consultants/architects break down the system into pieces that can be programmed." System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The system design specifications serve as input for the next phase of the model. Roles Involved: Architects & Team

Outcome: Technical Design Document (TDD)

2.3.4 **Coding** "The actual development starts and the product is built in coding phase. " The work is divided in modules/units and actual coding is started in this coding phase and it is the main focus for developer. Coding is one of the longest phase of SDLC. Roles involved: Developers and Architects Outcome: Programs or Application or Module

2.3.5 **Testing** "In Testing phase testers execute the test cases against the application, report the defects and retested the fixed defects. " During this phase unit testing, integration testing, system testing, acceptance testing are done. Roles Involved: Testers, Developers Outcome: Defects, Test Summary Report, Test Plan, Test Case document

2.3.6 **Delivery & Maintenance** "Delivery: After successful testing the product is delivered / deployed to the c During the Delivery phase, customer will perform user acceptance testing (UAT) in a real time environment. Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time. This process where the care is taken for the

developed product is known as maintenance. Roles Involved: Testers, Developers, Customer, Business team, Architects, Project Manager, and Delivery Manager Outcome: Quality Product, Enhancements &Production Issues (Maintenance)

3. **Software Development Life Cycle (SDLC) Models** "There are many development models have been developed in order to achieve different required objectives."

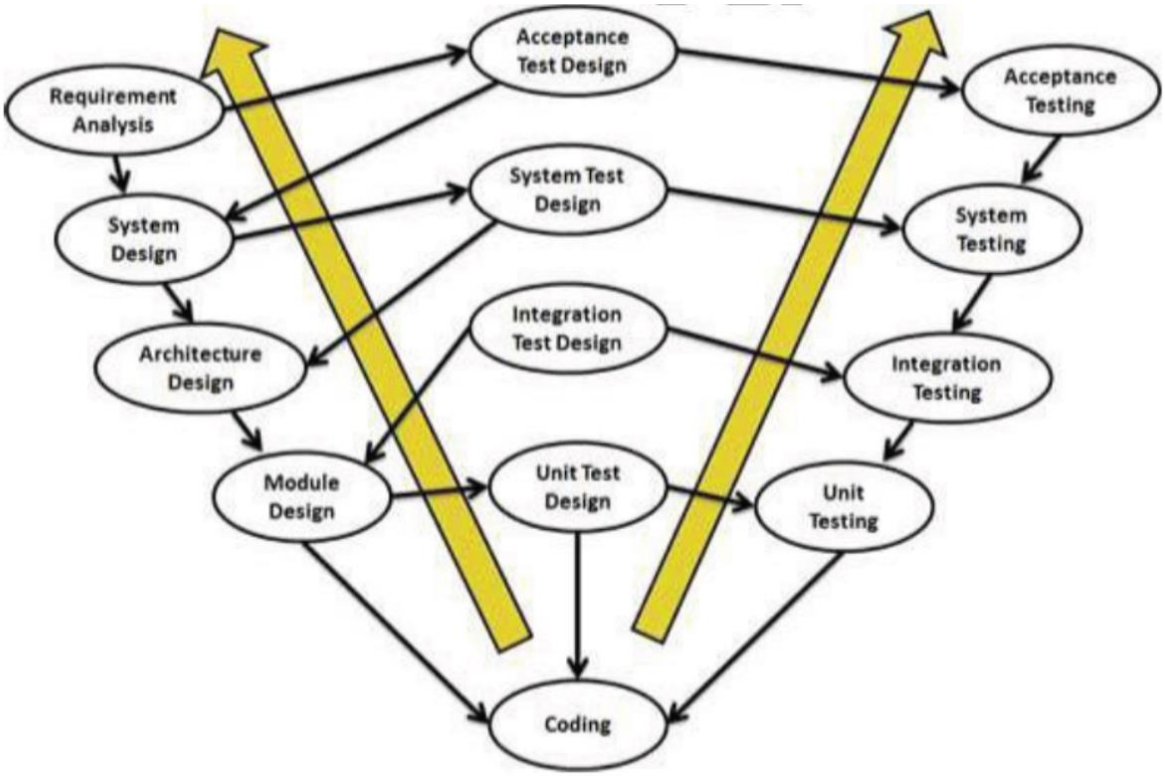
The selection of model has very high impact on the testing that is carried out. It will define what, where and when of our planned testing, influence regression testing and largely determines which test techniques to use.

3.1 **Water Fall Model** "In a waterfall model, each phase must be completed fully before the next phase ca The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. This type of model is basically used for the project which is small and there are no uncertain requirements.

Advantages of waterfall model: - This model is simple and easy to understand and use. - It is easy to manage due to the rigidity of the model - In this model phases are processed and completed one at a time.

Disadvantages of waterfall model: - Once an application is in the testing stage, it is very difficult to go back and change - No working software is produced until late during the life cycle. - High amounts of risk - Not a good model for complex and object-oriented projects. - Poor model for long and ongoing projects. When to use the waterfall model: - This model is used only when the requirements are very well known, clear and fixed. - The project is short.

3.2 **V Model**



"The - Vmodel is a SDLC model where execution of processes happens in a sequential manner in V-shape. " V-Shaped life cycle is a sequential path of execution of processes. Each phase must be completed before the next phase begins. Testing of the product is planned in parallel with a corresponding phase of development. **Advantages of V-model**: - Simple and easy to use. - Testing activities like planning, test designing happens well before coding. This saves a lot of time. - Proactive defect tracking – that is defects are found at early stage.

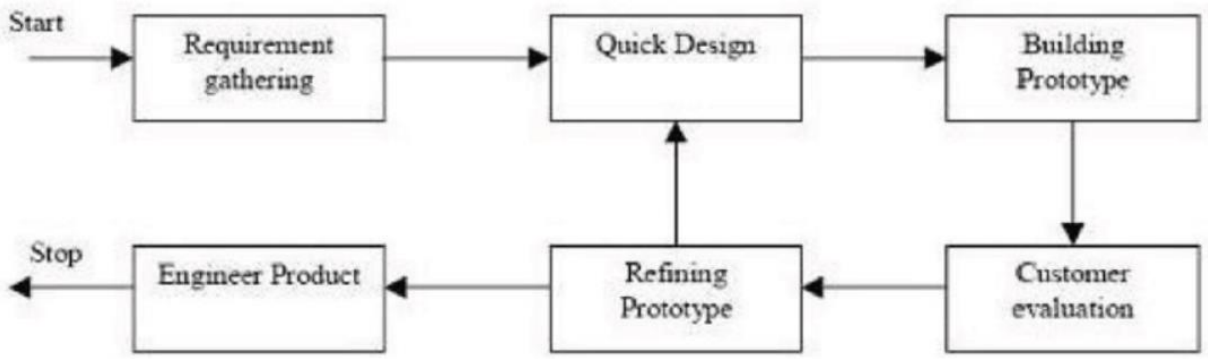
Disadvantages of V-model: - Software is developed during the implementation phase, so no early prototypes of the software are produced. - If any changes happen in midway, then the test documents along with requirement documents has to be updated.

When to use the V-model: -

The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed. - The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.

3.3 **Prototype Model**

Prototypes are the **first draft** in the product development phase before we reach the final product. When these prototypes are circulated to real-world users to find flaws and areas of improvement, it is known as prototype testing.



Prototyping Model

Advantages of Prototype model: - Users are actively involved in the development - Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.

- Errors can be detected much earlier.

- Quicker user feedback is available leading to better solutions.

- Missing functionality can be identified easily

- Confusing or difficult functions can be identified

Disadvantages of Prototype model:

- Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.

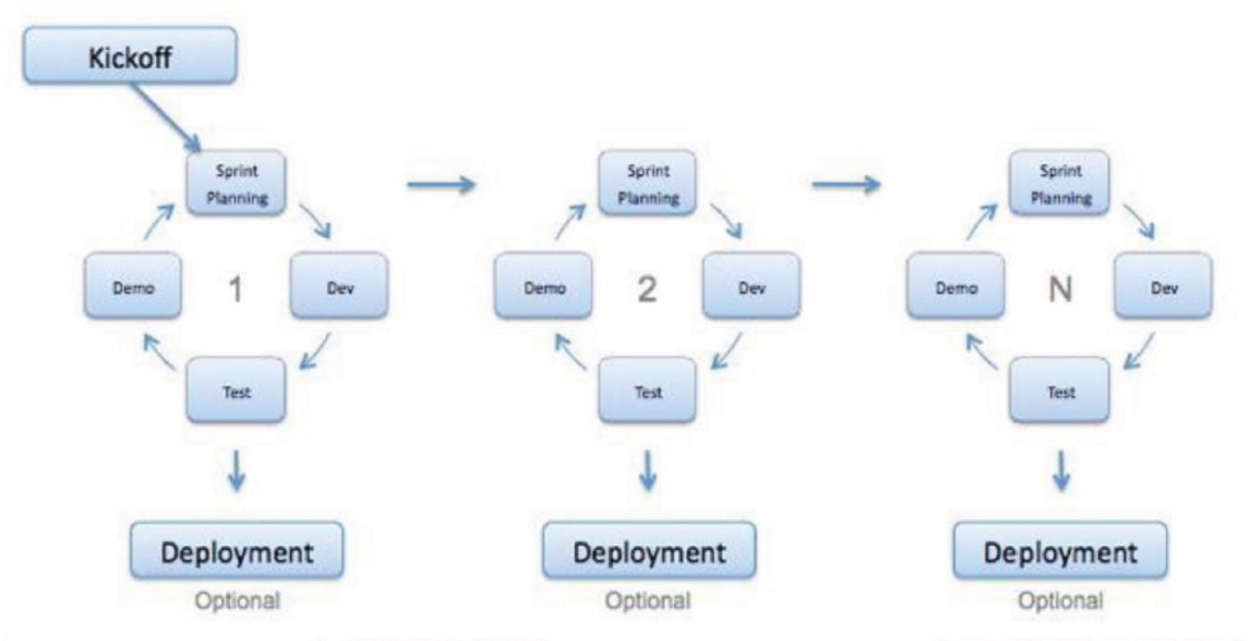
- Incomplete application may cause application not to be used as thefull system was designed

When to use Prototype model:

- Prototype model should be used when the desired system needs to have a lot of interaction with the end users.

3.4 **Agile Model**

Agile Testing is a testing practice that follows the rules and principles of agile software development. Unlike the Waterfall method, Agile Testing can begin at the start of the project with continuous integration between development and testing. Agile Testing methodology is not sequential (in the sense it's executed only after coding phase) but continuous.



Advantages of Agile model:

Agile Testing is a testing practice that follows the rules and principles of agile software development. Unlike the Waterfall method, Agile Testing can begin at the start of the project with continuous integration between development and testing. Agile Testing methodology is not sequential (in the sense it's executed only after coding phase) but continuous.

- Customer satisfaction by continuous delivery of software.

- Customers, developers and testers constantly interact with each other.

- Working software is delivered frequently

- Continuous attention to technical excellence and good design.

- Even late changes in requirements are welcomed

Disadvantages of Agile model:

- It is difficult to assess the effort required at the beginning of the software development life cycle.

- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.

- Only senior programmers are capable of taking the kind of decisions required during the development process.

When to use Agile model:

- New changes can be implemented at very little cost

- To implement a new feature the developers need to lose only the work of a few days.

- Unlike the waterfall model in agile model very limited planning is required to get started with the project.

Software Testing Methodologies

4.1 **Block Box Testing**

4.1.1 What is Block box testing

“The technique of testing without having technical knowledge of an application i Specification-based testing technique is also known as „black-box‟or input/output driven testing techniques because they view the software as a black-box with inputs and outputs.

4.1.2 How will perform block box testing

“Testing team will perform the block box testing” The testers have no knowledge of how the system or component is structured inside the box. In black-box testing the tester is concentrating on what the software does, not how it does it.

4.1.3 How to perform block box testing

Block box testing covers both functional and non-functional testing. Functional testing is concerned with what the system does its features or functions. Non-functional testing is concerned with examining how well the system does. Non-functional testing like performance,usability, portability, maintainability, etc. Advantages Disadvantages Well suited and efficient for large code segments. Limited Coverage since only a selected number of test scenarios are actually performed. Code Access not required. Inefficient testing, due to the fact that the tester only has limited Clearly separates user's perspective from the developer's knowledge about an application. perspective through visibly defined roles. Blind Coverage, since the tester cannot target specific code Large numbers of moderately skilled testers can test the segments or error prone areas. application with no knowledge of implementation, programming language or operating systems. The test cases are difficult to design.

4.1.4 Block box testing techniques

Below is the black box testing techniques: - Equivalence partitioning - Boundary value analysis - Error Guessing

4.2 White box testing

4.2.1 What is white box testing

“Structure-based testing technique is also known as „white-box‟ or „glass-box‟ testing technique because here testers require knowledge of how the software is implemented, how it works “

4.2.2 Who will perform White box testing

The developer will do the white box testing, and they will test line by line of code to find the bug. If they found any bug in any of the programs, they will correct it.

4.2.3 How to perform White box testing

White box testing is the detailed investigation of internal logic and structure of the code. White box testing is also called g lass testing or Openbox testing.

Advantages

- As the tester has knowledge of the source code, it becomes Due to the fact that a skilled tester is needed to perform white very easy to find out which type of data can help in testing the box testing, the costs are increased. application effectively.
- It helps in optimizing the code.
- Extra lines of code can be removed which can bring in hidden will go untested.
- Due to the tester's knowledge about the code, maximum specialized tools like code analyzers and debugging tools are required. coverage is attained during test scenario writing.

Disadvantages

- Sometimes it is impossible to look into every nook and corner to find out hidden errors
- It is difficult to maintain white box testing

4.3 **Grey Box testing**

4.3.1 What is grey box testing “Grey Box testing is a technique application with to limited test knowledge the of the internal working of an application.”

4.3.2 Who will perform Grey box testing

“Unlike blacktesting,boxwhere the tester only tests the application's user interface, in grey boxtesting, the tester has access to design documents and the database. Having this knowledge, the tester is able to better prepare test data and test scenarios when making the test plan.”

Advantages Disadvantages

Advantages	Disadvantages
Offers combined benefits of black box and white box testing wherever possible.	Since the access to source code is not available, the ability to go over the code and test coverage is limited.
Grey box testers don't rely on the source code; instead they rely on interface definition and functional specifications.	The tests can be redundant if the software designer has already run a test case.
Based on the limited information available, a grey box tester can design excellent test scenarios especially around communication protocols and data type handling.	Testing every possible input stream is unrealistic because it would take an unreasonable amount of time; therefore, many program paths will go untested.
The test is done from the point of view of the user and not the designer.	

5.2 **Module/Component Testing**

What is Module/Component Testing: “Component testing is a method where testing of each c application is done separately.” Suppose, in an application there are 5 components. Testing of each 5 components

separately and efficiently is called as component testing. Who will perform Component Testing:“Component testing is done by the tester.”

5.3 **Integration Testing**

What is Integration Testing:

“Integrations donetestingwhentwoi modules are integrated, in order to test the behavior and functionality of both the modules after integration.” As displayed in the image below when two different modules „Module A“ and „Modul integration testing is done.

Who will perform Integration Testing:

“Integration testing is done by a specific integration tester or test team.”

Integration Testing Techniques:

- Top down

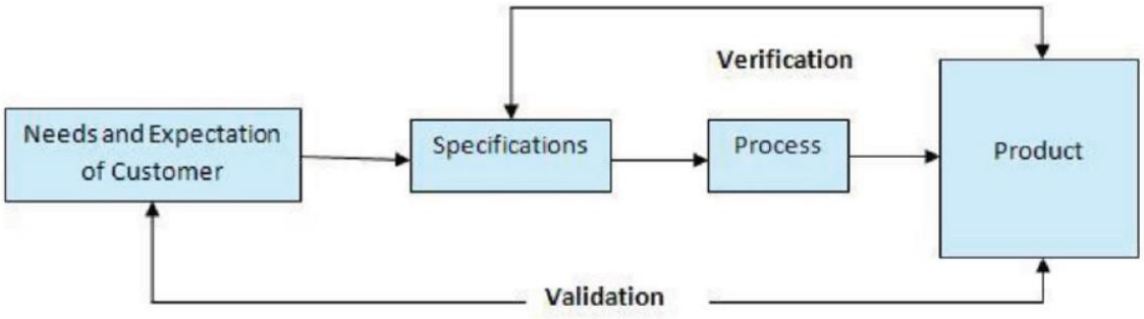
- Bottom Up

Top-down integration testing: Testing takes place from top to bottom, following the control flow or architectural. Components or systems are substituted by stubs. Below is the diagram of „Top down App roach“: Bottom-up integration testing: Testing takes place from the bottom of the control flow upwards. Components or systems are substituted by drivers. Below is the image of „Bottom up approach“:

Stub: A stub is called from the software component to be tested. Driver: A driver calls the component to be tested. 5.4 System Testing What is system testing: “Once all the components are integrated, the application as a wh see that it meets the requirements?” System testing is most often the final test to verify that the system to be delivered meets the specification and its purpose. Who will do the system testing: “System testing is carried out by specialist testers or inde

5.5 User Acceptance Testing (UAT) What is UAT: “Acceptance testing is basically done to ensurefth specificationthattherequirementsaremet. ”o - The goal of acceptance testing is to establish confidence in the system. - Acceptance testing is most often focused on a validation type testing. Who will perform UAT: “Acceptance testing is basically done by the user or customer although other stakeholders may be involved as well.” Alpha Testing: “Alpha testing is done at the developer‟s site. It is do ne at the end Beta testing: “Beta testing is done at the customers site. It is done just before the

6. Verification & Validation 6.1 Verification



6.1.1 What is Verification: “The process of evaluating work-products (not the actual final product) of a development phase to determine whether they meet the specified requirements Verification is done at the starting of the development process. It includes reviews and meetings, walkthroughs, inspection, etc. to evaluate documents, plans, code, requirements and specifications.

6.1.2 Who will perform: Peers (Sr Team Members, Architects, Analysts)

6.1.5 Walkthrough “A walkthrough is conducted by„documentheauthorunderof thereview‟ who takes the participants document and his or her thought processes, to achieve a common understanding and to gather feedback.”

6.2 Validation

6.1.1 What is Validation: “The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements.”

7. Functional & Non Functional Testing Functional Testing: “Functional Testing is a testing technique that is used to test the features/functionality of the Software”.

7.1 Smoke Testing:“Smoke testing refers to testing the basic functionality of the bui declared as unstable and it is NOT tested anymore until the smoke test of the build passes.

7.2 Sanity Testing:“Software testing technique performed by the test team for some basic tests. The aim of basic test is to be conducted whenever a new build is received for testing. “

The terminologies such as Smoke Test or Build Verification Test or Basic Acceptance Test or Sanity Test are interchangeably used, however, each one of them is used under a slightly different scenario. Sanity test is usually unscripted, helps to identify the dependent missing functionalities. It is used to determine if the section of the application is still working after a minor change.Sanity testing can be narrow and deep. Sanity test is a narrow regression test that focuses on one or a few areas of functionality

7.3 Re-testing: “Retesting is executing a previously failed test against new software to check if the problem is resolved. “

7.4 Regression Testing: “Regression testing is performed to verify if the build has NOT broken any other parts of the application by the recent code changes for defect fixing or for enhancement. “ The purpose of a regression testing is to verify that modifications in the software or the environment have not caused any unintended adverse side effects and that the system still meets its requirements.

7.5 Exploratory Testing: “Testing of software without any documents (test cases or test planning) and Identify the functionality of application by exploring the application and exploring & learni

7.6 Monkey Testing: “Monkey testing is a software testing technique in which the testi system under test randomly. The Input data that is used to test also generated r

7.7 End to End Testing:“End-to-end testing is a methodology used to test whether the flow of an application is performing as designed from start to finish.” The purpose of carrying out end-to-end tests is to identify system dependencies and to ensure that the right information is passed between various system components and systems. Difference between System testing and End to End testing: There isn't really a huge difference between the two and in some establishments the terms could be used interchangeably. Everywhere is different. System testing: You're testing the whole system i.e. all of its components to ensure that each is functioning as intended. This is more from a functional side to check against requirements. End to end testing: This is more about the actual flow through a system in a more realistic end user scenario. Can a user navigate the application as expected and does it work. You're testing the workflow. Non Functional Testing:

7.8 User Interface Testing:“Graphical User Interface (GUI) testing is checking the application design of an application”. Ex: Required/Optional, Fields Align, Lengths, Progress Bars, Scroll Bars, Alignments, etc

7.9 Usability Testing: “In usability testing basically the testers tests the ease with which the user interfaces can be used. It tests that whether the application is user-friendly or not. “ Usability Testing tests the following features of the software. – How easy it is to use the software. – How easy it is to learn the software. – How convenient is the software to end user.

7.10 Stress Testing: “It is a form of testing that is used to determine the stability of a given system, Stress testing involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results. “ Stress testing is a generic term used to describe the process of putting a system through stress.

7.11 Load Testing: “Load testing is performed to determine a system’s behavior under both normal and at peak conditions. “ A load test is usually conducted to understand the behavior of the application under a specific expected load. E.g. If the number of users are in creased then how much CPU, memory will be consumed, what is the network and bandwidth response time.

7.12 Performance Testing: “Performance testing is testing that is performed, to determine how fast some aspect of a system performs under a particular workload. “ It can serve different purposes like it can demonstrate that the system meets performance criteria.

7.13 Localization Testing: “Localization translates the product UI and occasionally changes some initial settings to make it suitable for another region.” Localization testing checks the quality of a product's localization for a particular target culture/locale. The test effort during localization testing focuses on: - Areas affected by localization, such as UI and content - Culture/locale-specific, language-specific, and region-specific areas

7.14 Globalization Testing: “Globalization Testing is testing process to check whether software can perform properly in any locale or culture & functioning properly with all types of international inputs and steps to effectively make your product truly global

This type of testing validates whether the application is capable for using all over the world and to check whether the input accepts all the language texts. Ex: Let’s see another example of a Zip code field in Sign up form: - For globalized, it should allow to enter alphanumeric inputs -

For localized (country like INDIA), it should allow only numbers in input field.

7.15 Security Testing: “Security testing is basically to check that whether the application or the product is secured or not. “ Can anyone came tomorrow and hack the system or login the application without any authorization. It is a process to determine that an information system protects data and maintains functionality as intended. Security testing is related to the security of data and the functionality of the application. You should be aware of the following concepts while performing security testing: 1. Confidentiality - The application should only provide the data to the relevant party e.g. one customer's transactional data should not be visible to another customer; the irrelevant personal details of the customer should not be visible to the administrator and so on. 2. Integrity - The data stored and displayed by the application should be correct e.g. after a withdrawal, the customer's account should be debited by the correct amount. 3. Authentication - It should be possible to attribute the data transmitted in the application to either the application or the customer. In other words, no one other than the customer or the bank should be able to create or modify any data. 4. Authorization - The application or a user should only be able to perform the tasks which they are respectively authorized to perform e.g. a customer should not be able to withdraw more than the balance in their account without having an overdraft facility, the application should not be able to levy charges on a customer account without prior customer approval. 5. Availability - The data and functionality should be available to the users throughout the working period e.g. if the bank's operating times are from 8 a.m. to 8 p.m. on all working days, it should be possible for a customer to access their account and make the necessary transactions on their account. 6. Non-repudiation - At a later date, it should not be possible for a party to deny that a particular transaction or data change took place e.g. if a customer withdraws an amount from their account, this should trigger

the relevant actions (posting to their transaction records, debiting their account and sending them a notification etc.).

7.16 Compatibility Testing: ”Compatibility Testing ensure compatibility of the application built with various other objects such as other web browsers, hardware platforms, operating systems etc.” This type of testing helps find out how well a system performs in a particular environment that includes hardware, network, operating system and other software etc. Ex: Browser Compatibility Testing, OS Compatibility Testing

7.17 Installation Testing: “Installation testing is performed to ensure that all necessary components are installed properly and working as per the requirements of the software, post installation. Installation process may include partial, full or upgrade install. “

7.18 Recovery Testing: “Recovery testing is done in order to check how fast and better the application can recover after it has gone through any type of crash or failure“ Ex: For example, when an application is receiving data from a network, unplug the connecting cable. After some time, plug the cable back in and analyze the application’s ability to continue receiving data from the point at which the network connection got disappeared. Restart the system while a browser has a definite number of sessions and check whether the browser is able to recover all of them or not. 8. Windows & Web Application

8.1 What is Windows application: “A program that is written to run under Microsoft's Windows operating system. “

8.2 What is Web application: “Web application is an application that is accessed via Web browser over a network such as the Internet or an intranet. “

9. Software Testing Life Cycle (STLC) “STLC consists of series of activities carried out methodologically to help certify your software product.

These activities are part ofthe Software Testing Life Cycle.” The different stages in Software Test Life Cycle:

9.1 Requirement Analysis: “During this phase, test team studies the requirements from a testing point of view to identify the testable requirements.” The QA team may interact with various stakeholders (Client, Business Analyst, Technical Leads, and System Architects etc.) to understand the requirements in detail. Requirements could be either Functional (defining what the software must do) or Non Functional (defining system performance /security availability).

Deliverables:Requirement Traceability Matrix (RTM), Clarification Document RTM: “Traceability matrix links a business requirement to its corresponding functional requirement.” If a Test Case fails, traceability helps determine the corresponding functionality easily; it also helps ensure that all requirements are tested. Sample RTM

Clarification Document: “Contains all clarifications which will arise during the requirement analysis phase”. Sample: Sno Date Module/Page Name Query Description Comments 9.2 Test Planning: What is Test plan: “Test planning is the first step of the testing process. In this phase we identify the activities and resources which would help to meet the testing objectives.” What Test Plan contains: (IEEE 829 STANDARD TEST PLAN TEMPLATE) Test plan identifier Test deliverables Introduction Test tasks Test items Environmental needs Features to be tested Responsibilities Features not to be tested Staffing and training needs Approach Schedule Item pass/fail criteria Risks and contingencies Who will prepare the test plan: Test Lead or Test Manager.

What is Test Strategy:“A Test Strategy document is a high level document and normally developed by project manager. This document defines “Software Testing Approach” to achieve testing objectives. The Test Strategy is normally derived from the Business Requirement Specification document.” Test plan Vs Test Strategy: Generally it doesn’t matter which comes first. Test planning document is a

combination of strategy plugged with overall project plan. According to IEEE Standard 829-2008, strategy plan is a sub item of test plan. Every organization has their own standards and processes to maintain these documents. Some organizations include strategy details in test plan itself. Some organizations list strategy as a subsection in testing plan but details is separated out in different test strategy document. Ex:Test plan gives the information of who is going to test at what time. For example: Module 1 is going to be tested by “X tester”. If tester Y replaces X for some reason, the test plan has to be updated. On the contrary, test strategy is going to have details like – “Individual modules are to be tested by test team members. “ In this case, it does not matter who is testing it- so it’s generic and the change in the team member does not have to be updated, keeping it static. Deliverables: Test Plan with estimation 9.3 Test Case Development: “During this phase the test cases will be prepared”.

9.3.1 What is test case: “A test case is a set of conditions under which a tester will determine whether a system under test satisfies requirements or works correctly

9.3.4 Types of test cases Functional Test Cases: “The test cases based on functional requirement specifications” Positive Test Cases: “Test Cases with valid input and also verifying that the outputs are correct.” Negative Test Cases:“This testing involves exercising application functionality using a combination of invalid inputs, some unexpected operating conditions and by some other “out-of-bounds” scenarios.” Non Functional Test Cases:“The test cases based on functional requirement specifications like performance, Load, Stress, Security, etc.”

9.3.5 Test Case Review: Reviewing is a form of testing too – the verification part of the V&V, also called static testing. Why Review: For exactly the same reason we test the software • To uncover errors • To check for completeness • To make sure the standards and guidelines

are followed Review Checklist: • Do test cases cover all requirements? • Has each test case been assigned a test case identifier? • Does each test case specify? - Actions - Test condition - Expected result • Have the expected results been recorded in detail? • Is any method for validating expected results specified? • Do test cases for field validations, record validations and database updates include the following? - Valid conditions - Invalid conditions - Boundary or unusual conditions • Do the test cases for reports include the test data along with the expected output? • Have the inter test case dependencies been described? • Have Pass/Fail criteria been specified?

• Have all requested environments been specified? • Has the method for logging on to the test environment been specified? • Are pre-conditions for the test specified? • Is the number of Test cases met customer standards? Self-Review: “Review our own work by us” Peer Review: “Review our own work by colleague” Lead Review: “Review our work by our Test lead or Manager” Client Review: “Review our work by client or business team after lead review.” Deliverables: Test Case Document, Test Data

9.4 Test Environment Setup “Test environment decides the software and hardware conditions under which a work product is tested. “ Test environment set-up is one of the critical aspects of testing process and can be done in parallel with Test Case Development Stage. Test team may not be involved in this activity if the customer or dev team provides the test environment in which case the test team is required to do a readiness check (smoke testing) of the given environment. Deliverables: Test Environment How many environments do we have: “A Typical project can have following environments” - Dev - QA - Pre-Production - Production 9.5 Test Execution “In this phase testing team start executing test cases based on prepared test planning & prepared test cases in the prior step in testing environment”. Deliverables: Test cases updated with results

9.6 Result Analysis & Reporting: “After test execution phase, if the test case is passed then same can be marked as Passed. If any test case is failed then corresponding defect can be reported to developer team via bug tracking system & bug can be linked for corresponding test case for further analysis.” Severity:It is the extent to which the defect can affect the software. In other words it defines the impact that a given defect has on the system. If an application or web page crashes when a remote link is clicked, in this case clicking the remote link by an user is rare but the impact of application crashing is severe. So the severity is high but priority is low.

• Critical: The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable and there is no acceptable alternative method to achieve the required results then the severity will be stated as critical.

• Major: The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable but there exists an acceptable alternative method to achieve the required results then the severity will be stated as major.

• Medium: The defect that does not result in the termination, but causes the system to produce incorrect, incomplete or inconsistent results then the severity will be stated as moderate. • Minor: The defect that does not result in the termination and does not damage the usability of the system and the desired results can be easily obtained by working around the defects then the severity is stated as minor

. • Cosmetic: The defect that is related to the enhancement of the system where the changes are related to the look and field of the application then the severity is stated as cosmetic. Priority: “Priority defines the order in which we should resolve a defect. Should we fix it

now, or can it wait?" This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mentioned then the developer has to fix it at the earliest. The priority status is set based on the customer requirements. For example: 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.

- Low: The defect is an irritant which should be repaired, but repair can be deferred until after more serious defect has been fixed.

- Medium: The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created.

- High: The defect must be resolved as soon as possible because the defect is affecting the application or the product severely. The system cannot be used until the repair has been done. Few very important scenarios related to the severity and priority which are asked during the interview:

- 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. (Eg. A site maintaining the student details, on saving record if it, doesn't allow to save the record then this is high priority and high severity bug.)

- High Priority & Low Severity: The spelling mistakes that happens on the cover page or heading or title of an application.

- High Severity & Low Priority: An error which occurs on the functionality of the application (for which there is no workaround) and will not allow the user to use the system but on click of link which is rarely used by the end user.

• Low Priority and Low Severity: Any cosmetic or spelling issues which is within a paragraph or in the report (Not on cover page, heading, title). Famous Defect Tracking Tools: - Bugzilla - JIRA - ALM (QC)

A typical Defect Tracker will have: - Defect id - Date - Created By - Assigned TO - Bug description - Steps to Reproduce - Expected Result - Comments - Screen shots Defect Life Cycle:“Defect life cycle is a cycle which a defect goes through during its lifetime. “ It starts when defect is found and ends when a defect is closed, after ensuring it’s not reproduced. Defect life cycle is related to the bug found during testing. The Life cycle of the bug can be shown diagrammatically as follows:

• New: When a defect is logged and posted for the first time. It’s state is given as new.

• Assigned: After the tester has posted the bug, the lead of the tester approves that the bug is genuine and he assigns the bug to corresponding developer and the developer team. It’s state given as assigned. • Open: At this state the developer has started analyzing and working on the defect fix.

• Fixed: When developer makes necessary code changes and verifies the changes then he/she can make bug status as ‘Fixed’ and the bug is passed to testing team.

• Pending retest: After fixing the defect the developer has given that particular code for retesting to the tester. Here the testing is pending on the testers end. Hence its status is pending retest. • Retest: At this stage the tester do the retesting of the changed code which developer has given to him to check whether the defect got fixed or not.

• Verified: The tester tests the bug again after it got fixed by the developer. If the bug is not present in the software, he approves that the bug is fixed and changes the status to “verified”.

- Reopen: If the bug still exists even after the bug is fixed by the developer, the tester changes the status to “reopened”. The bug goes through the life cycle once again.

- Closed: Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, he changes the status of the bug to “closed”. This state means that the bug is fixed, tested and approved.

- Duplicate: If the bug is repeated twice or the two bugs mention the same concept of the bug, then one bug status is changed to “duplicate“.

- Rejected: If the developer feels that the bug is not genuine, he rejects the bug. Then the state of the bug is changed to “rejected”.

- Deferred: The bug, changed to deferred state means the bug is expected to be fixed in next releases. The reasons for changing the bug to this state have many factors. Some of them are priority of the bug may be low, lack of time for the release or the bug may not have major effect on the software.

- Not a bug: The state given as “Not a bug” if there is no change in the functionality of the application. For an example: If customer asks for some change in the look and field of the application like change of color of some text then it is not a bug but just some change in the looks of the application.

9.7 Delivery & Maintenance Evaluate cycle completion criteria based on Time,Test overage,Cost,Software,Critical Business Objectives, Quality. Typical Exit criteria may include the following:

- All Test plans have been run.

- A certain level of requirements coverage has been achieved.

- No high priority or severe bugs are left outstanding.

- All high-risk areas have been fully tested, with only minor residual risks left outstanding.

Cost – when the budget has been spent.

- The schedule has been achieved. Maintenance: Once a system is deployed it is in service for years and decades. During this time the system and its operational environment is often corrected, changed or extended. Testing that is provided during this phase is called maintenance testing. Usually maintenance testing is consisting of two parts: First one is, testing the changes that has been made because of the correction in the system or if the system is extended or because of some additional features added to it. Second one is regression tests to prove that the rest of the system has not been affected by the maintenance work. Metrics: A Metric is a quantitative measure of the degree to which a system, system component, or process possesses a given attribute. Metrics can be defined as "STANDARDS OF MEASUREMENT". Software Metrics are used to measure the quality of the project. Simply, Metric is a unit used for describing an attribute. Metric is a scale for measurement. Suppose, in general, "Kilogram" is a metric for measuring the attribute "Weight". Similarly, in software, "How many issues are found in thousand lines of code?", here No. of issues is one measurement & No. of lines of code is another measurement. Metric is defined from these two measurements. Test metrics example: • How many defects are existed within the module?

- How many test cases are executed per person?

- What is the Test coverage %? Why Test Metrics? Generation of Software Test Metrics is the most important responsibility of the Software Test Lead/Manager. "We cannot improve what we cannot measure." "We cannot control what we cannot measure"

- Take decision for next phase of activities

- Evidence of the claim or prediction
- Understand the type of improvement required
- Take decision on process or technology change Effectiveness: Doing the right thing. It deals with meeting the desirable attributes that are expected by the customer. Efficiency: Doing the thing right. It concerns the resources used for the service to be rendered a. Test Plan coverage on Functionality: Formula: (No of requirements covered / total number of requirements)

* 100 b. Test Case defect density: Formula: (Defective Test Scripts /Total Test Scripts)

* 100 Example: Total test script developed 1360, total test script executed 1280, total test script passed 1065, total test script failed 215. So, test case defect density is :215 X 100 / 1280 = 16.8% This 16.8% value can also be called as test case efficiency %, which is depends upon total number of test cases which uncovered defects. c. Defect Slippage Ratio: Number of defects slipped (reported from production) v/s number of defects reported during execution. Formula: Number of Defects Slipped / (Number of Defects Raised - Number of Defects Withdrawn) Example: Customer filed defects are 21, total defect found while testing are 267, total number of invalid defects are 17. So, Slippage Ratio is [21/ (267-17)] X 100 = 8.4%

d. Requirement Volatility: Number of requirements agreed v/s number of requirements changed. Formula: Number of Requirements Added + Deleted + Modified) *100 / Number of Original Requirements Example: VSS 1.3 release had total 67 requirements initially, later they added another 7 new requirements and removed 3 from initial requirements and modified 11 requirements. So, requirement Volatility is (7 + 3 + 11) * 100/67 = 31.34% Means almost 1/3 of the requirement changed after initial identification e. Review Efficiency: The Review Efficiency is a metric that offers insight on the review quality and testing some

organization also use this term as “Static Testing” efficiency and they are aiming to get min of 30% defects in static testing. Formula=100*Total number of defects found by reviews/Total number of project defects Example: A project found total 269 defects in different reviews, which were fixed and test team got 476 defects which were reported and valid So, Review efficiency is [269/(269+476)] X 100 = 36.1% f. Defect Removal Effectiveness(DRE): DRE= (Defects removed during development phase x100%) / Defects latent in the product Defects latent in the product = Defects removed during development Phase+ defects found later by

Terminology

1. Project: Requirements will come from one customer and mostly it will be used by customer and his people.

2. Product: Requirements will come from various customers and will be used by more number of customers.

3. Application: Group of programs designed for the customers to use for specific operations.

4. AUT (Application under Testing): After Designing and coding phase of development cycle,the application(build) comes for testing then the application(build) is called "Application Under test".

5. Quality: Justification of the Requirements and absence of Defects, delivered on time.

6. Defect: Deviation from the requirement

7. SRS: System Requirement Specification, which is the actual requirement document.

8. BDD: Business Design Document, which is an initial document designed by business people and then they will prepare the SRS.

9. Mock ups: In Design phase or for requirement phase, some sample screens replica of actual application will be provided for team. 10. Use Case: It will describe the Basic Flow, Alternate Flow and Exceptional Flow, how the application will be processed. Use case document will be used as the base document by both the Developers and the Testers. Developers will write the code based on the Use case and Testers in turn will identify test scenarios and write test cases based on the use case. Use will be written by the Business Analysts from the Functionality requirement document.

11. Test Case: Test cases are the step wise description or activities which are going to be executed in order to validate the application. The test cases will contain the step number, description or activity - what action is going to happen while validation, the input data, the expected result and the actual result. The test cases are written based on the functionality or requirements or the use cases received from the client. Test Cases will be prepared by Testing Team.

12. Test data: Test data is the data that is used in tests of a software system. In order to test a software application you need to enter some data for testing most of the features. Any such specifically identified data which is used in tests is known as test data. You can have test data in excel sheet which can be entered manually while executing test cases or it can be read automatically from files (XML, Flat Files, Database etc.) by automation tools.

13. Version Control: It is a system that records changes to a file or set of files over time so that you can recall specific versions later.

14. Check out: Taking the document to edit (do the changes, ex: Updating the defect tracker, test cases, etc) from the configuration management tool or common repository (VSS).

15. Check in: Releasing the document after performing the changes to the configuration management tool or common repository (VSS).

16. Release notes: Release notes is a document which will be prepared by the dev team during the release time. It will be delivered to the customers which contains the technical information about changes addressed in the current release.

17. Build: Build is a version of software will be given for testing team to test. Usually release contains multiple builds to test. Software is still in testing.

18. Release: Software is no longer in testing phase, ready to deliver to customer.

19. Delivery: Handover the working software constructed based on the customer requirements to the customer on or before delivery date.

20. Test Deliverables: The documents which were prepared during the testing phase. - Clarification Document / Query Tracker - Traceability Matrix - Test Case Document with results - Defect Report (Open, Closed, Defer) - Test Summary Report (Passed, Failed, Hold)

22. Traceability Matrix: t is a document that maps and traces user requirement with test cases. The main purpose of Requirement Traceability Matrix is to see that all test cases are covered so that no functionality should miss while testing.

23. Productivity: Effective hours produced by individual team members in the team. Ex, if a person stays at office more than 10 hrs and he works only for 7 hrs and 3 hrs he didn't work on the project, Then the productivity is 7 hrs.

24. Variance: Variance is the difference between what was planned and the actual value. Ex, if planned date is 20th Jan 15 and actual completion date is 25th Jan 15 then the variance is 5 days.

25. Estimation: Providing the time to finish the specific work.

26. Escalation: Taking the issue to next step. If there is a defect and defect is not accepting by the dev team but still you believe that it's an issue, you can contact customer for the issue.

27. Roles & Responsibilities: The work which we are going to perform after getting the Job. The Responsibilities will change based on role. Ex, Test engineer will do requirement understanding, test case preparation, execution and reporting the bugs. Test Lead will assign the work to his team members, collects data, etc.

28. Entry Criteria: Entry criterion is used to determine when a given test activity should start. It also includes the beginning of a level of testing, when test design or when test execution is ready to start.

29. 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.

30. Status Call: Status call will happen daily / weekly / monthly based on client/company rules where we will discuss about project status.

31. MOM: Minutes of Meeting are the written or recorded documentation that is used to inform attendees and non-attendees of the happenings during the meeting. MOM Contains: The names of the participants, the agenda items covered, decisions made by the participants, the follow-up actions committed to by participants, due dates for the completion of commitments , and any other events or discussions worth documenting for future review or history.

32. One to One: It is a face to face meeting.

33. Appraisal: The comp people (HR, Managers, and team) will assess the individual team member based on Peer/Lead/Manager feedback and they will revise your salary or role. There will be Midterm and annual appraisals based on company rules.

34. Rating: Giving ranking for the individual team member by the Lead/managers based on their skills and their performance during appraisal period.

37. Non Billable: A resource that is not being charged from client is called Shadow / non billable resource.

38. OOO: Out of office, going out from office. Usually uses while informing to team that you are going out from office.

39. Planned Leave: You planned your leave long before and applied. It's a scheduled or planned well before.

40. Un Planned or Sick Leave: Unscheduled and unplanned leave, usually take if resource is not feeling well. Usually we do have 16 planned leave (4 per quarter) and 5-10 sick leaves. It's diff per company.

41. PTO: Paid Time Off, taking leave. Using the leaves (Planned Leave, Sick Leave) and still company pay you salary for this time.

42. Notice Period: Once you want to move out of from the current company then we will quit the Job. Company has a policy that we will have to inform before 2/3 months so that they will search for a replacement for us.

43. Bench Mark: Finalising any document or any process. Ex, Finalising the test cases, requirement.

44. CTC: Cost to Company (CTC) is the salary package of an employee. It indicates the total amount of expense an employer (organization) is spending for an employee in a year.

45. Variable Pay: Variable pay is used generally to recognize and reward employee contribution toward company productivity, profitability, team work, safety, quality, or some other metric deemed important. This will be pay you based on your performance and company performance during the appraisal period.

46. Basic Salary: Basic salary is the amount paid to an employee before any extras are added or taken off, such as reductions because of salary sacrifice schemes or an increase due to overtime or a

bonus. Allowances, such as internet for home-based workers or contributions to phone usage, would also be added to the basic salary.

47. Gross Salary: Gross salary is the term used to describe all of the money you've made while working at your job, figured before any deductions are taken for state and federal taxes, Social Security and health insurance. If you work more than one job, you'll have a gross salary amount for each one.

48. Demo: Demonstrating the completed part of work or software to customer and will get his feedback.

49. POC: Point of Contact, to whom we need to contact. There will be diff for diff departments. Ex, for all QA related queries, you can contact Mr John, for all development related queries you can contact Mr Anderson. (OR) Proof of Concept: A proof of concept (POC) is a demonstration, the purpose of which is to verify that certain concepts or theories have the potential for real-world application. POC is therefore a prototype that is designed to determine feasibility, but does not represent deliverables. Proof of concept is also known as proof of principle.

50. Pipeline: In progress, expecting something shortly or near future. Ex, there is more work in pipe line.

51. Project Sign in: Project was finalised and the team is ready to work on.

INTERVIEW FAQ'S Q.

What is the MAIN benefit of designing tests early in the life cycle?

Ans: It helps prevent defects from being introduced into the code. Q. What is risk-based testing? Ans: Risk-based testing is the term used for

an approach to creating a test strategy that is based on prioritizing tests by risk. The basis of the approach is a detailed risk analysis and prioritizing of risks by risk level. Tests to address each risk are then specified, starting with the highest risk first.

Q. A wholesaler sells printer cartridges. The minimum order quantity is 5. There is a 20% discount for orders of 100 or more printer cartridges. You have been asked to prepare test cases using various values for the number of printer cartridges ordered. Which of the following groups contain three test inputs that would be generated using Boundary Value Analysis?

Ans: 4, 5, 99 Q.

What is the KEY difference between preventative and reactive approaches to testing?

Ans: Preventative tests are designed early; reactive tests are designed after the software has been produced.

Q. What is the purpose of exit criteria?

Ans: The purpose of exit criteria is to define when a test level is completed.

Q. What determines the level of risk?

Ans: The likelihood of an adverse event and the impact of the event determine the level of risk.

Q. What is the MAIN objective when reviewing a software deliverable?

Ans: To identify defects in any software work product.

Q. Which of the following defines the expected results of a test? Test case specification or test design specification.

Ans: Test case specification defines the expected results of a test.

Q: What is the benefit of test independence?

Ans: It avoids author bias in defining effective tests.

Q. As part of which test process do you determine the exit criteria?

Ans: The exit Criteria is determined on the bases of ‘Test Planning’.

Q. What is beta testing?

Ans: Testing performed by potential customers at their own locations.

Q. What is the difference between Testing Techniques and Testing Tools?

Ans: Testing technique: – Is a process for ensuring that some aspects of the application system or unit functions properly there may be few techniques but many tools. Testing Tools: Is a vehicle for performing a test process. The tool is a resource to the tester, but it is insufficient to conduct testing

Q. We use the output of the requirement analysis, the requirement specification as the input for writing … Ans: User Acceptance Test Cases

Q. Repeated testing of an already tested program, after modification, to discover any defects introduced or uncovered as a result of the changes in the software being tested or in another related or unrelated software component:

Ans: Regression Testing

Q. What is component testing?

Ans: Component testing, also known as unit, module and program testing, searches for defects in, and verifies the functioning of software (e.g. modules, programs, objects, classes, etc.) that are separately testable. Component testing may be done in isolation from the rest of the system depending on the context of the development life cycle and the system. Most often stubs and drivers are used to replace the missing software and simulate the interface between the software

components in a simple manner. A stub is called from the software component to be tested; a driver calls a component to be tested.

Q. What is functional system testing?

Ans: Testing the end to end functionality of the system as a whole is defined as a functional system testing.

Q. What are the benefits of Independent Testing?

Ans: Independent testers are unbiased and identify different defects at the same time.

Q. What are the different Methodologies in Agile Development Model?

Ans: There are currently seven different agile methodologies that I am aware of:

1. Extreme Programming (XP)

2. Scrum

3. Lean Software Development

4. Feature-Driven Development

5. Agile Unified Process

6. Crystal

7. Dynamic Systems Development Model (DSDM)

Q: Which activity in the fundamental test process includes evaluation of the testability of the requirements and system?

Ans: A 'Test Analysis' and 'Design' includes evaluation of the testability of the requirements and system.

Q: What is typically the MOST important reason to use risk to drive testing efforts? Ans: Because testing everything is not feasible.

Q. What is random/monkey testing? When it is used?

Ans: Random testing often known as monkey testing. In such type of testing data is generated randomly often using a tool or automated mechanism. With this randomly generated input the system is tested and results are analysed accordingly. These testing are less reliable; hence it is normally used by the beginners and to see whether the system will hold up under adverse effects.

Q. Which of the following are valid objectives for incident reports?

Ans: 1. Provide developers and other parties with feedback about the problem to enable identification, isolation and correction as necessary. 2. Provide ideas for test process improvement. 3. Provide a vehicle for assessing tester competence. 4. Provide testers with a means of tracking the quality of the system under test.

Q: Consider the following techniques. Which are static and which are dynamic techniques?

Ans:

1. Equivalence Partitioning.

2. Use Case Testing.

3. Data Flow Analysis.

4. Exploratory Testing.

5. Decision Testing.

6. Inspections. Data Flow Analysis and Inspections are static; Equivalence Partitioning, Use Case Testing, Exploratory Testing and Decision Testing are dynamic.

Q: Why are static testing and dynamic testing described as complementary?

Ans: Because they share the aim of identifying defects but differ in the types of defect they find.

Q: What are the phases of a formal review?

Ans: In contrast to informal reviews, formal reviews follow a formal process.

A typical formal review process consists of six main steps:

1. Planning

2. Kick-off

3. Preparation

4. Review meeting

5. Rework

6. Follow-up.

Q: What is the role of moderator in review process?

Ans: The moderator (or review leader) leads the review process. He or she determines, in co-operation with the author, the type of review, approach and the composition of the review team. The moderator performs the entry check and the follow-up on the rework, in order to control the quality of the input and output of the review process. The moderator also schedules the meeting, disseminates documents before the meeting, coaches other team members, paces the meeting, leads possible discussions and stores the data that is collected.

Q: What is an equivalence partition (also known as an equivalence class)?

Ans: An input or output ranges of values such that only one value in the range becomes a test case.

Q: When should configuration management procedures be implemented?

Ans: During test planning.

Q: A Type of functional Testing, which investigates the functions relating to detection of threats, such as virus from malicious outsiders?

Ans: Security Testing

Q: Testing wherein we subject the target of the test , to varying workloads to measure and evaluate the performance behaviours and ability of the target and of the test to continue to function properly under these different workloads?

Ans: Load Testing

Q: Testing activity which is performed to expose defects in the interfaces and in the interaction between integrated components is?

Ans: Integration Level Testing

Q: What are the Structure-based (white-box) testing techniques?

Ans: Structure-based testing techniques (which are also dynamic rather than static) use the internal structure of the software to derive test cases. They are commonly called 'white-box' or 'glass-box' techniques (implying you can see into the system) since they require knowledge of how the software is implemented, that is, how it works. For example, a structural technique may be concerned with exercising loops in the software. Different test cases may be derived to exercise the loop once, twice, and many times. This may be done regardless of the functionality of the software.

Q: When “Regression Testing” should be performed?

Ans: After the software has changed or when the environment has changed Regression testing should be performed.

Q: What is negative and positive testing?

Ans: A negative test is when you put in an invalid input and receives errors. While a positive testing, is when you put in a valid input and

expect some action to be completed in accordance with the specification.

Q: What is the purpose of a test completion criterion?

Ans: The purpose of test completion criterion is to determine when to stop testing

Q: What can static analysis NOT find?

Ans: For example memory leaks.

Q: What is the difference between re-testing and regression testing?

Ans: Re-testing ensures the original fault has been removed; regression testing looks for unexpected side effects.

Q: What are the Experience-based testing techniques?

Ans: In experience-based techniques, people's knowledge, skills and background are a prime contributor to the test conditions and test cases. The experience of both technical and business people is important, as they bring different perspectives to the test analysis and design process. Due to previous experience with similar systems, they may have insights into what could go wrong, which is very useful for testing.

Q: What type of review requires formal entry and exit criteria, including metrics? Ans: Inspection Q: Could reviews or inspections be considered part of testing?

Ans: Yes, because both help detect faults and improve quality.

Q: An input field takes the year of birth between 1900 and 2004 what are the boundary values for testing this field?

Ans: 1899, 1900, 1950, 1951, 2004, 2005

Q: What is the one Key reason why developers have difficulty testing their own work?

Ans: Lack of Objectivity

Q: When should testing be stopped?

Ans: It depends on the risks for the system being tested. There are some criteria bases on which you can stop testing.

1. Deadlines (Testing, Release)

2. Test budget has been depleted

3. Bug rate fall below 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

Q: Which of the following is the main purpose of the integration strategy for integration testing in the small?

Ans: The main purpose of the integration strategy is to specify which modules to combine when and how many at once.

Q: What is black box testing? What are the different black box testing techniques?

Ans: Black box testing is the software testing method which is used to test the software without knowing the internal structure of code or program. This testing is usually done to check the functionality of an application. The different black box testing techniques are 1. Equivalence Partitioning 2. Boundary value analysis

Q: Which review is normally used to evaluate a product to determine its suitability for intended use and to identify discrepancies?

Ans: Technical Review.

Q: Faults found should be originally documented by whom?

Ans: By testers.

Q: Which is the current formal world-wide recognized documentation standard?

Ans: There isn't one.

Q: Which of the following is the review participant who has created the item to be reviewed?

Ans: Author

Q: A number of critical bugs are fixed in software. All the bugs are in one module, related to reports. The test manager decides to do regression testing only on the reports module.

Ans: Regression testing should be done on other modules as well because fixing one module may affect other modules.

Q: Why does the boundary value analysis provide good test cases?

Ans: Because errors are frequently made during programming of the different cases near the 'edges' of the range of values.

Q: What makes an inspection different from other review types?

Ans: It is led by a trained leader, uses formal entry and exit criteria and checklists.

Q: Why can be tester dependent on configuration management?

Ans: Because configuration management assures that we know the exact version of the testware and the test object.

Q: What is a V-Model?

Ans: A software development model that illustrates how testing activities integrate with software development phases

Q: What is maintenance testing?

Ans: Triggered by modifications, migration or retirement of existing software

Q: What is test coverage?

Ans: Test coverage measures in some specific way the amount of testing performed by a set of tests (derived in some other way, e.g. using specification-based techniques). Wherever we can count things and can tell whether or not each of those things has been tested by some test, then we can measure coverage

Q: When do we prepare RTM (Requirement traceability matrix), is it before test case designing or after test case designing?

Ans: It would be before test case designing. Requirements should already be traceable from Review activities since you should have traceability in the Test Plan already. This question also would depend on the organisation. If the organisations do test after development started then requirements must be already traceable to their source. To make life simpler use a tool to manage requirements.

Q: What is called the process starting with the terminal modules?

Ans: Bottom-up integration

Q: During which test activity could faults be found most cost effectively?

Ans: During test planning

Q. The purpose of requirement phase is

Ans: To freeze requirements, to understand user needs, to define the scope of testing

Q. Why we split testing into distinct stages?

Ans: We split testing into distinct stages because of following reasons,

1. Each test stage has a different purpose

2. It is easier to manage testing in stages

3. We can run different test into different environments

4. Performance and quality of the testing is improved using phased testing

Q. What is DRE?

Ans: To measure test effectiveness a powerful metric is used to measure test effectiveness known as DRE (Defect Removal Efficiency) From this metric we would know how many bugs we have found from the set of test cases. Formula for calculating DRE is DRE=Number of bugs while testing / number of bugs while testing + number of bugs found by user

Q. Which of the following is likely to benefit most from the use of test tools providing test capture and replay facilities?

a) Regression testing

b) Integration testing

c) System testing

d) User acceptance testing

Ans: Regression testing

Q. How would you estimate the amount of re-testing likely to be required?

Ans: Metrics from previous similar projects and discussions with the development team

Q. What studies data flow analysis?

Ans: The use of data on paths through the code.

Q. What is Alpha testing?

Ans: Pre-release testing by end user representatives at the developer's site.

Q. What is a failure?

Ans: Failure is a departure from specified behaviour.

Q. What are Test comparators?

Ans: Is it really a test if you put some inputs into some software, but never look to see whether the software produces the correct result? The essence of testing is to check whether the software produces the correct result, and to do that, we must compare what the software produces to what it should produce. A test comparator helps to automate aspects of that comparison.

Q. Who is responsible for document all the issues, problems and open point that were identified during the review meeting

Ans: Scribe

Q. What is the main purpose of Informal review

Ans: Inexpensive way to get some benefit

Q.What is the purpose of test design technique?

Ans: Identifying test conditions and Identifying test cases

Q. When testing a grade calculation system, a tester determines that all scores from 90 to 100 will yield a grade of A, but scores below 90 will not. This analysis is known as:

Ans:Equivalence partitioning

Q. A test manager wants to use the resources available for the automated testing of a web application. Ans:The best choice is Tester, test Automater, web specialist, DBA

Q. During the testing of a module tester ‘X’ finds a bug and assigned it to developer. But developer rejects the same, saying that it’s not a bug. What ‘X’ should do?

Ans:Send to the detailed information of the bug encountered and check the reproducibility

Q. In practice, which Life Cycle model may have more, fewer or different levels of development and testing, depending on the project and the software product. For example, there may be component integration testing after component testing, and system integration testing after system testing.

Ans: V-Model

Q. Which technique can be used to achieve input and output coverage? It can be applied to human input, input via interfaces to a system, or interface parameters in integration testing.

Ans: Equivalence partitioning

Q.This life cycle model is basically driven by schedule and budget risks” This statement is best suited for…

Ans: V-Model

Q. In which order should tests be run?

Ans: The most important one must tests first

Q. The later in the development life cycle a fault is discovered, the more expensive it is to fix. Why?

Ans: The fault has been built into more documentation, code, tests, etc

Q. What is Coverage measurement?

Ans: It is a partial measure of test thoroughness.

Q. What is Boundary value testing?

Ans: Test boundary conditions on, below and above the edges of input and output equivalence classes. For instance, let say a bank application where you can withdraw maximum Rs.20,000 and a minimum of Rs.100, so in boundary value testing we test only the exact boundaries, rather than hitting in the middle. That means we test above the maximum limit and below the minimum limit.

Q.The purpose of which is allow specific tests to be carried out on a system or network that resembles as closely as possible the environment where the item under test will be used upon release?

Ans: Test Environment

Q. What can be thought of as being based on the project plan, but with greater amounts of detail?

Ans: Phase Test Plan

Q. What is exploratory testing?

Ans: Exploratory testing is a hands-on approach in which testers are involved in minimum planning and maximum test execution. The planning involves the creation of a test charter, a short declaration of the scope of a short (1 to 2 hour) time-boxed test effort, the objectives and possible approaches to be used. The test design and test execution activities are performed in parallel typically without formally documenting the test conditions, test cases or test scripts. This does not mean that other, more formal testing techniques will not be used. For example, the tester may decide to use boundary value analysis but will think through and test the most important boundary values without necessarily writing them down. Some notes will be written during the exploratory-testing session, so that a report can be produced afterwards.

Q. What is “use case testing”?

Ans: In order to identify and execute the functional requirement of an application from end to finish “use case” is used and the techniques used to do this is known as “Use Case Testing” Bonus!

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

Ans: The complete Verification and Validation of software is done in SDLC, while STLC only does Validation of the system. SDLC is a part of STLC.

Q. What is traceability matrix? Ans: The relationship between test cases and requirements is shown with the help of a document. This document is known as traceability matrix.

Q. What is Equivalence partitioning testing?

Ans: Equivalence partitioning testing is a software testing technique which divides the application input test data into each partition at least once of equivalent data from which test cases can be derived. By this testing method it reduces the time required for software testing.

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

Ans: White box testing technique involves selection of test cases based on an analysis of the internal structure (Code coverage, branches coverage, paths coverage, condition coverage etc.) of a component or system. It is also known as Code-Based testing or Structural testing. Different types of white box testing are 1. Statement Coverage 2. Decision Coverage

Q. In white box testing what do you verify?

Ans: In white box testing following steps are verified.

1. Verify the security holes in the code

2. Verify the incomplete or broken paths in the code

3. Verify the flow of structure according to the document specification

4. Verify the expected outputs

5. Verify all conditional loops in the code to check the complete functionality of the application

6. Verify the line by line coding and cover 100% testing

Q. What is the difference between static and dynamic testing?

Ans: Static testing: During Static testing method, the code is not executed and it is performed using the software documentation. Dynamic testing: To perform this testing the code is required to be in an executable form.

Q. What is verification and validation?

Ans: Verification is a process of evaluating software at development phase and to decide whether the product of a given application satisfies the specified requirements. Validation is the process of evaluating software at the end of the development process and to check whether it meets the customer requirements.

Q. What are different test levels?

Ans:There are four test levels 1. Unit/component/program/module testing

2. Integration testing

3. System testing

4. Acceptance testing

Q. What is Integration testing?

Ans: Integration testing is a level of software testing process, where individual units of an application are combined and tested. It is usually performed after unit and functional testing.

Q. What are the tables in testplans?

Ans: 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 is the difference between UAT (User Acceptance Testing) and System testing?

Ans: System Testing: System testing is finding defects when the system under goes testing as a whole, it is also known as end to end testing. In such type of testing, the application undergoes from beginning till the end. UAT: User Acceptance Testing (UAT) involves running a product through a series of specific tests which determines whether the product will meet the needs of its users.

Q. What is Dynamic Testing?

Ans. It is the testing done by executing the code or program with various input values and output is verified.

Q. What is GUI Testing?

Ans. GUI or Graphical user interface testing is the process of testing software user interface against the provided requirements/ Mock-ups/HTML designs.

Q. What is Formal Testing? Ans. Software verification carried out by following test plan, testing procedures and proper documentation with an approval from customer

Q. What is Risk Based Testing?

Ans: Identifying the critical functionality in the system then deciding the orders in which these functionality to be tested and applying testing.

Q. What is Early Testing?

Ans. Conducting testing as soon as possible in development life cycle to find defects at early stages of SDLC. Early testing is helpful to reduce the cost of fixing defects at later stages of STLC.

Q. What is Exhaustive Testing?

Ans. Testing functionality with all valid, invalid inputs and preconditions is called exhaustive testing

. Q. What is Defect Clustering?

Ans. Any small module or functionality may contain more number of defects – concentrate more testing on this functionality.

Q. What is Static Testing? Ans. Manual verification of the code without executing the program is called as static testing. In this process issues are identified in code by checking code, requirement and design documents.

Q. What is Positive Testing? Ans. Testing conducted on the application to determine if system works. Basically known as “test to pass” approach.

Q. What is Negative Testing? Ans: Testing Software with negative approach to check if system is not “showing error when not supposed to” and “not showing error when supposed to”.

Q. What is End-to-End Testing? Ans. Testing the overall functionality of the system including the data integration among all the modules is called end to end testing.

Q. What is Exploratory Testing? Ans. Exploring the application, understanding the functionality, adding (or) modifying existing test cases for better testing is called exploratory testing.

Q. What is Monkey Testing

Ans. Testing conducted on a application without any plan and carried out with tests here and there to find any system crash with an intention of finding tricky defects is called monkey testing.

Q. What is Non-functionality Testing?

Ans. Validating various non-functional aspects of the system such as user interfaces, user friendliness security, compatibility, Load, Stress and Performance etc is called non-functional testing.

Q. What is Usability Testing?

Ans. Checking how easily the end users are able to understand and operate the application is called Usability Testing.

Q. What is Security Testing

Ans. Validating whether all security conditions are properly implemented in the software (or) not is called security Testing.

Q. What is Performance Testing?

Ans. Process of measuring various efficiency characteristics of a system such as response time, through put, load stress transactions per minutes transaction mix.

Q. What is Load Testing?

Ans. Analysing functional and performance behaviour of the application under various conditions is called Load Testing.

Q. What is Stress Testing?

Ans. Checking the application behaviour under stress conditions (OR) Reducing the system resources and keeping the load as constant checking how does the application is behaving is called stress testing.

Q. What is Process?

Ans. A process is set of a practices performed to achieve a give purpose; it may include tools, methods, materials and or people.

Q. What is Software Configuration Management?

Ans. The Process of Identifying, Organizing and controlling changes to software development and maintenance. (OR) A methodology to control and manage a software development project

Q. What is Testing Process / Life Cycle?

Ans. Write Test Plan

Test Scenarios Test Cases Executing Test Cases Test Results Defect Reporting Defect Tracking Defect Closing Test Release

Q. What is full form of CMMI?

Ans. Capability Maturity Model Integration

Q. What is Code Walk Through?

Ans. Informal analysis of the program source code to find defects and verify coding techniques.

Q. What is Unit Level Testing?

Ans. Testing of single programs, modules or unit of code.

Q. What is Integration Level Testing?

Ans. Testing of related programs, Modules (or) Unit of code. (OR) Partitions of the system ready for testing with other partitions of the system.

Q. What is System Level Testing?

Ans. Testing of entire computer system across all modules. This kind of testing can include functional and structural testing.

Q. What is Alpha Testing?

Ans. Testing of whole computer system before rolling out to the UAT.

Q. What is User Acceptance Testing (UAT)?

Ans. Testing of computer system by client to verify if it adhered to the provided requirements.

Q. What is Test Plan?

Ans. A document describing the scope, approach, resources, and schedule of testing activities. It identifies test items, features to be tested, testing tasks, who will do each task, and any risks requiring contingency planning.

Q. What is Test Scenario?

Ans. Identify all the possible areas to be tested (or) what to be tested.

Q. What is a Defect?

Ans.Expected result is not matching with the application actual result.

Q. What is Severity?

Ans. It defines the important of defect with respect to functional point of view i.e. how critical is defect with respective to the application.

Q. What is Priority?

Ans. It indicates the importance or urgency of fixing a defect

Q. What is Re-Testing?

Ans. Retesting the application to verify whether defects have been fixed or not.

Q. What is Regression Testing?

Ans. Verifying existing functional and non-functional area after making changes to the part of the software or addition of new features.

Q. What is Recovery Testing?

Ans. Checking if the system is able to handle some unexpected unpredictable situations is called recovery testing.

Q. What is Globalization Testing?

Ans. Process of verifying software whether it can be run independent of its geographical and cultural environment. Checking if the application is having features of setting and changing language, date, format and currency if it is designed for global users.

Q. What is Localization Testing?

Ans. Verifying of globalized application for a particular locality of users, cultural and geographical conditions.

Q. What is Installation Testing?

Ans. Checking if we are able to install the software successfully (or) not as per the guidelines given in installation document is called installation testing.

Q. What is Un-installation Testing?

Ans. Checking if we are able to uninstall the software from the system successfully (or) not is called Uninstallation Testing

Q. What is Compatibility Testing?

Ans. Checking if the application is compatible to different software and hardware environment or not is called compatibility testing.

Q. What is Test Strategy?

Ans. It is a part of test plan describing how testing is carried out for the project and what testing types needs to be performed on the application.

Q. What is Test Case?

Ans. A Test case is a set of preconditions steps to be followed with input data and expected behaviour to validate a functionality of a system.

Q. What is Business Validation Test Case?

Ans. A test case is prepared to check business condition or business requirement is called business validation test case.

Q. What is a Good Test Case?

Ans. A Test case that have high priority of catching defects in called a good test case.

Q. What is Use Case Testing?

Ans. Validating a Software to confirm whether it is developed as per the use cases or not is called use case testing.

Q. What is Defect Age?

Ans. The time gap between date of detection & date of closure of a defect.

Q. What is Showstopper Defect?

Ans. A defect which is not permitting to continue further testing is called Showstopper Defect

Q. What is Test Closure?

Ans. It is the last phase of the STLC, Where the management prepares various test summary reports that explains the complete statistics of the project based on the testing carried out.

Q. What is Bucket Testing?

Ans. Bucket testing is also Know as A/B testing. It is mostly used to study the impact of the various product designs in website metrics. Two simultaneous versions are run on a single or set of web pages to measure the difference in click rates, interface and traffic.

Q. What is Concurrency Testing?

Ans. This is a multiple user testing to access the application at the same time to verify effect on code, module or DB. Mainly used to identify locking and deadlocking situations in the code.

Q. What is Web Application Testing?

Ans. Web application testing is done on a website to check – load, performance, security, Functionality, Interface, Compatibility and other usability related issues.

Q. What is Unit Testing?

Ans. Unit testing is done to check whether the individual modules of the source code are working properly or not.

Q. What is Interface Testing

Ans. Interface testing is done to check whether the individual modules are communicating properly as per specifications. Interface testing is mostly used to test the user interface of GUI applications.

Q. What is Test Harness?

Ans. Test Harness is configuring a set of tools and test data to test an application in various conditions, which involves monitoring the output with expected output for correctness. The benefits of Testing Harness

are: Productivity increase due to process automation and increase in product quality

Q. What is Scalability Testing?

Ans. It is used to check whether the functionality and performance of a system, whether system is capable to meet the volume and size changes as per the requirements Scalability testing is done using load test by changing various software, hardware configurations and testing environment.

Q. What is Fuzz Testing?

Ans. Fuzz testing is a black box testing technique which uses a random bad data to attack a program to check if anything breaks in the application.

Q. What is Difference between QA, QC and testing?

Ans.

QA? It is process oriented Aim is to prevent defects in an application

QC? Set of activities used to evaluate a developed work product It is product oriented Testing? Executing and verifying application with the intention of finding defects

Q. What is Date Driven Testing?

Ans. It is Automation testing process in which application is tested with multiple set of data with different preconditions as an input to the script.