Module–2(Manual Testing)

1) What is Exploratory Testing?

Exploratory testing is a concurrent process where Test design, execution and logging happen simultaneously, Testing is often not recorded & Makes use of experience, heuristics and test patterns. Testing is based on a test charter that may include Scope of the testing (in and out), The focus of exploratory testing is more on testing as a "thinking" activity.

Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking. Automation has its limits. Its not random testing but it is Adhoc testing with purpose of find bugs.

2) What is traceability matrix?

Test conditions should be able to be linked back to their sources in the test basis, this is known as traceability. Traceability can be horizontal through all the test documentation for a given test level (e.g. system testing, from test conditions through test cases to test scripts) or it can be vertical through the layers of development documentation (e.g. from requirements to components).
To protect against changes you should be able to trace back from every system component to the original requirement that caused its presence. A software process should help you keeping the virtual table up-to-date.

3) What is Boundary value testing?

Boundary value analysis is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges. Boundary value analysis is a method which refines equivalence partitioning. Boundary value analysis generates test cases that highlight errors better than equivalence partitioning. Boundary Value Analysis (BVA) uses the same analysis of partitions as EP and is usually used in conjunction with EP in test case design.

4) What is Equivalence partitioning testing?

Aim is to treat groups of inputs as equivalent and to select one representative input to test them all**.** EP can be used for all Levels of Testing
Equivalence partitioning is the process of defining the optimum number of tests by: Reviewing documents such as the Functional Design Specification and Detailed Design Specification, and identifying each input condition within a function, Selecting input data that is representative of all other data that would likely invoke the same process for that particular condition.
The numbers fall into a partition where each would have the same, or equivalent, result i.e. an Equivalence Partition (EP) or Equivalence Class. If one value finds a bug, the others probably will too. If one doesn't find a bug, the others probably

won't either. In EP we must identify Valid Equivalence partitions and Invalid Equivalence partitions where applicable (typically in range tests). The Valid partition is bounded by the values 1 and 100.

5) What is Integration testing?

Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. Integration Testing is a level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems. Integration testing is done by a specific integration tester or test team. Components may be code modules, operating systems, hardware and even complete systems.
There are 2 levels of Integration Testing

- Component Integration Testing
- System Integration Testing

6) What determines the level of risk?

A properly designed test that passes, reduces the overall level of Risk in a system. Risk – 'A factor that could result in future negative consequences; usually expressed as impact and likelihood'. When testing does find defects, the Quality of the software system increases when those defects are fixed.
When testing does find defects, the Quality of the software system increases when those defects are fixed. The Quality of systems can be improved through Lessons learned from previous projects.
Analysis of root causes of defects found in other projects can lead to Process Improvement. Process Improvement can prevent those defects reoccurring.
Risks should be evaluated at the Business Level, Technological Level, Project Level and Testing Level. Risks are also used to decide where to start testing and where more testing is needed. Risk analysis should be used to determine what to test in each component and just as importantly what not to test.

7) What is Alpha testing?

Alpha Testing is definitely performed and carried out at the developing organizations location with the involvement of developers. It is always performed by the developers at the software development site. Sometimes it is also performed by Independent Testing Team. Alpha Testing is not open to the market and public. It is conducted for the software application and project. It is always performed in Virtual Environment. It is always performed within the organization. It is the form of Acceptance Testing. It comes under the category of both White Box Testing and Black Box Testing.

8) What is beta testing?

Beta Testing (field testing) is performed and carried out by users or you can say people at their own locations and site using customer data. It is always performed by the customers at their own site. It is not performed by Independent Testing Team. Beta Testing is always open to the market and public. It is usually conducted for software product. It is performed in Real Time Environment. It is always performed outside the organization. It is also the form of Acceptance Testing. It is only a kind of Black Box Testing.

9) What is component testing?

A minimal software item that can be tested in isolation. It means "A unit is the smallest testable part of software." Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed. Unit testing is the first level of testing and is performed prior to Integration Testing. Sometimes known as Unit Testing, Module Testing or Program Testing. Component can be tested in isolation – stubs/drivers may be employed. Unit testing frameworks, drivers, stubs and mock or fake objects are used to assist in unit testing. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended with debugging tool. Unit testing is performed by using the White Box Testing method.

10) What is functional system testing?

A requirement that specifies a function that a system or system component must perform. A Requirement may exist as a text document and/or a model.
There is two types of Test Approach.

- Requirement Based Functional Testing.
- Process Based Testing

Functional System Testing Functionality As below:

1. Accuracy: Provision of right or agreed results or effects
2. Interoperability: Ability to interact with specified systems
3. Compliance: Adhere to applicable standards, conventions, regulations or laws
4. Auditability: Ability to provide adequate and accurate audit data
5. Suitability: Presence and appropriateness of functions for specified tasks

11) What is Non-Functional Testing?

Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portability. Non-functional testing includes, but is not limited to, performance testing, load testing, stress testing, usability testing, maintainability testing, reliability testing and portability testing. It is the testing of "how" the system works. Non-functional testing may be performed at all test levels. The term non-functional testing

describes the tests required to measure characteristics of systems and software that can be quantified on a varying scale, such as response times for performance testing.

12) What is GUI Testing?

Graphical User Interface (GUI) testing is the process of testing the system's GUI of the System under Test. GUI testing involves checking the screens with the controls like menus, buttons, icons, and all types of bars – tool bar, menu bar, dialog boxes and windows etc.

Approach of GUI Testing

a) **MANUAL BASED TESTING:** Under this approach, graphical screens are checked manually by testers in conformance with the requirements stated in business requirements document.
b) **MANUAL BASED TESTING**: Under this approach, graphical screens are checked manually by testers in conformance with the requirements stated in business requirements document.
c) **MODEL BASED TESTING:** A model is a graphical description of system's behavior. It helps us to understand and predict the system behavior. Models help in a generation of efficient test cases using the system requirements.

13) What is Adhoc testing?

Adhoc testing is an informal testing type with an aim to break the system. It does not follow any test design techniques to create test cases. In fact is does not create test cases altogether! This testing is primarily performed if the knowledge of testers in the system under test is very high. Adhoc Testing does not follow any structured way of testing and it is randomly done on any part of application. Main aim of this testing is to find defects by random checking. Adhoc testing can be achieved with the testing technique called Error Guessing. Error guessing can be done by the people having enough experience on the system to "guess" the most likely source of errors. The Error guessing is a technique where the experienced and good testers are encouraged to think of situations in which the software may not be able to cope. This is why an error guessing approach, used after more formal techniques have been applied to some extent, can be very effective.

14) What is load testing?

Its a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.

Load testing is a kind of performance testing which determines a system's performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously

This testing usually identifies –

The maximum operating capacity of an application. Determine whether current infrastructure is sufficient to run the application. Sustainability of application with respect to peak user load. Number of concurrent users that an application can support, and scalability to allow more users to access it. It is a type of non-functional testing. Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

15) What is stress Testing?

System is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load. Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions. It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions. Stress Testing is done to make sure that the system would not crash under crunch situations. Stress testing is also known as endurance testing. Under Stress Testing, AUT is be stressed for a short period of time to know its withstanding capacity. Most prominent use of stress testing is to determine the limit, at which the system or software or hardware breaks.

16) What is white box testing and list the types of white box testing?

Testing based on an analysis of the internal structure of the component or system. Structure-based testing technique is also known as 'white-box' or 'glass-box' testing technique because here the testers require knowledge of how the software is implemented, how it works. In white-box testing the tester is concentrating on how the software does it. Structure-based techniques are also used in system and acceptance testing, but the structures are different. White box testing is the detailed investigation of internal logic and structure of the code. White box testing is also called glass testing or open box testing. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code.

Types of White Box Testing

- Path Testing
- Loop Testing
- Conditional Testing
- Unit Testing
- Mutation Testing
- Integration Testing
- Penetration Testing

17) What is black box testing? What are the different black box testing techniques?

Testing, either functional or non-functional, without reference to the internal structure of the component or system. 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. 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. The technique of testing without having any knowledge of the interior workings of the application is Black Box testing. Typically, when performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

Techniques of Black Box Testing

1. Equivalence partitioning
2. Boundary value analysis
3. Decision tables
4. State transition testing
5. Use-case Testing

18) Mention what are the categories of defects?

a) **Data Quality/Database Defects**: Deals with improper handling of data in the database.
b) **Critical Functionality Defects**: The occurrence of these bugs hampers the crucial functionality of the application. Examples: - Exceptions
c) **Functionality Defects**: These defects affect the functionality of the application.
d) **Security Defects**: Application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix.
e) **User Interface Defects**: As the name suggests, the bugs deal with problems related to UI are usually considered less severe.

19) Mention what bigbang testing is?

In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole. Big Bang testing has the advantage that everything is finished before integration testing starts. The major disadvantage is that in general it is time consuming and difficult to trace the cause of failures because of this late integration. Here all component are integrated together at once, and then tested.

20) What is the purpose of exit criteria?

Exit Criteria defines the items that must be completed before testing can be concluded. In an Ideal world, you will not enter the next stage until the exit criteria for the previous stage is met. But practically this is not always possible.
Purpose of exit criteria is to define when we STOP testing either at the:

a) End of all testing – i.e. product Go Live
b) End of phase of testing (e.g. hand over from System Test to UAT)

Exit criteria is used to determine when testing at any stage is complete The set of generic and specific conditions, agreed upon with the stakeholders, for permitting a process to be officially completed
Exit criteria may be defined in terms of :
Thoroughness – i.e. coverage or requirements
cost or time constraints

percentage of tests run without incident
number of faults remaining

21) When should "Regression Testing" be performed?

Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed. when the system is stable and the system or the environment changes. when testing bug-fix releases as part of the maintenance phase. It should be applied at all Test Levels.
It should be considered complete when agreed completion criteria for regression testing have been met. Regression test suites evolve over time and given that they are run frequently are ideal candidates for automation.

22) What is 7 key principles? Explain in detail?

1) **Testing shows presence of Defects**: Testing can show that defects are present, but cannot prove that there are no defects. Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness. As we find more defects, the probability of undiscovered defects remaining in a system reduces. However Testing cannot prove that there are no defects present.
2) **Exhaustive Testing is Impossible! :** Testing everything including all combinations of inputs and preconditions is not possible. So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts. This is very unlikely that the project timescales would allow for this number of tests. So, accessing and managing risk is one of the most important activities and reason for testing in any project. We have learned that we cannot test everything (i.e. all combinations of inputs and pre-conditions).
3) **Early Testing :** Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives. Testing activities should start as early as possible in the development life cycle. These activities should be focused on defined objectives – outlined in the Test Strategy.
4) **Defect Clustering :** A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures. Defects are not evenly spread in a system. They are 'clustered'. In other words, most defects found during testing are usually confined to a small number of modules. Similarly, most operational failures of a system are usually confined to a small number of modules.
5) **Pesticide Paradox:** If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects. To overcome this "pesticide paradox", the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects. Testing identifies bugs, and programmers respond to fix them As bugs are eliminated by the programmers, the software improves As software improves the effectiveness of previous tests erodes

6) **Testing is Context Dependent:** Testing is Context Dependent. Testing is done differently in different contexts. Whilst, Testing can be 50% of development costs, in NASA's Apollo program it was 80% testing.
7) **Absence of Errors Fallacy:** If the system built is unusable and does not fulfill the user's needs and expectations then finding and fixing defects does not help. If we build a system and, in doing so, find and fix defects . It doesn't make it a good system. Even after defects have been resolved it may still be unusable and/or does not fulfil the users' needs and expectations.

23) Difference between QA v/s QC v/s Tester

Sr.No	Quality Assurance	Quality Control	Testing
1	Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements.	Activities which ensure the verification of developed software with respect to documented (or not in some cases) requirements.	Activities which ensure the identification of bugs/error/defects in the Software.
2	Focuses on processes and procedures rather than conducting actual testing on the system.	Focuses on actual testing by executing Software with intend to identify bug/defect through implementation of procedures and process.	Focuses on actual testing.
3	Process oriented activities.	Product oriented activities.	Product oriented activities.
4	Preventive activities.	It is a corrective process.	It is a preventive process.
5	It is a subset of Software Test Life Cycle (STLC).	QC can be considered as the subset of Quality Assurance.	Testing is the subset of Quality Control.

24) Difference between Smoke and Sanity?

Sr.No	Smoke Testing	Sanity Testing
1	Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine	Sanity Testing is done to check the new functionality / bugs have been fixed
2	The objective of this testing is to verify "stability" of the system in order to proceed with more rigorous testing	The objective of the testing is to verify the the "rationality" of the system in order proceed with more rigorous testing.
3	This testing is performed by the developers or testers	Sanity testing is usually performed by testers
4	Smoke testing is usually documented or scripted	Sanity testing is usually not documented and is unscripted

5	Smoke testing is a subset of Regression testing	Sanity testing is a subset of Acceptance testing
6	Smoke testing exercises the entire system from end to end	Sanity testing exercises only the particular component of the entire system
7	Smoke testing is like General Health Check Up	Sanity Testing is like specialized health check up

25) Difference between verification and Validation

Criteria	Verification	Validation
Defination	The process of evaluating work-products (not the actual final product) of a development phase to determine whether they meet the specified requirements for that phase.	The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements.
Objective	To ensure that the product is being built according to the requirements and design specifications. In other words, to ensure that work products meet their specified requirements	To ensure that the product actually meets the user's needs, and that the specifications were correct in the first place. In other words, to demonstrate that the product fulfills its intended use when placed in its intended environment.
Evaluation Items	Plans, Requirement Specs, Design Specs, Code, Test Cases	The actual product/software.
Activities	Reviews · Walkthroughs · Inspections	Testing

26) Explain types of Performance testing.

- **Load testing:-** Load testing is a kind of performance testing which determines a system's performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.
- **Stress testing:-** Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions.
- **Endurance testing:-** Endurance Testing is non-functional type of software testing where a software is tested with high load extended over a significant amount of time to evaluate the behavior of software application under sustained use.
- **Spike testing:-** Spike testing is a type of performance testing in which an application receives a sudden and extreme increase or decrease in load. The goal of spike testing is to determine the behavior of a software application when it receives extreme variations in traffic.
- **Volume testing:-** Volume testing comes under software testing. It helps us to check the behavior of an application by inserting a massive volume of the load in terms of data is known as volume testing. In volume testing, we will concentrate on the number of data rates than the number of users.

- **Scalability testing:-** A scalability test is a type of load testing that measures the application's ability to scale up or down as a reaction to an increase in the number of users. In other words, it tests how the system is going to perform during a sudden spike or fall of user request loads.

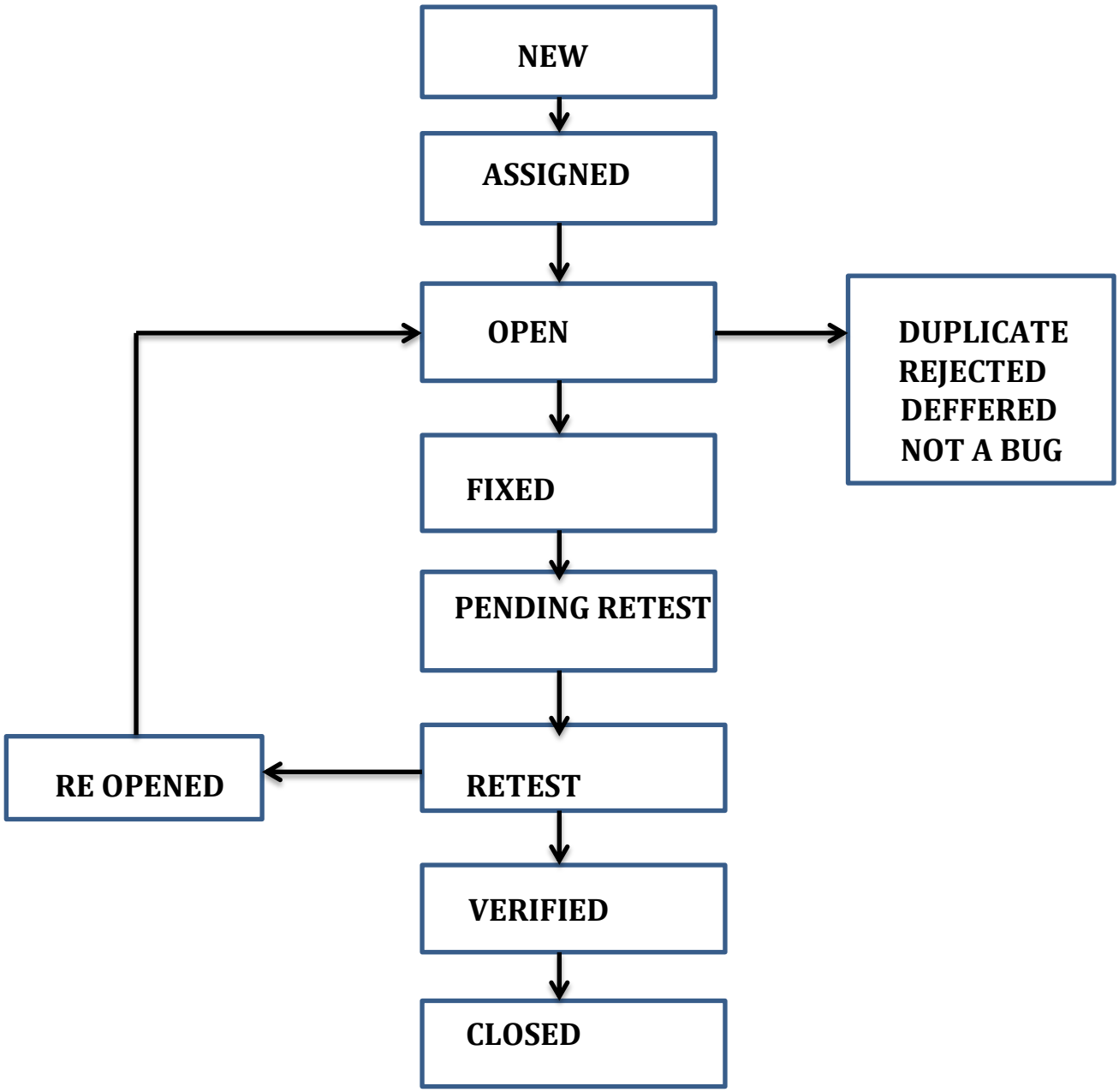
27) What is Error, Defect, Bug and failure?

"A mistake in coding is called error, error found by tester is called defect, defect accepted by development team then it is called bug, build does not meet the requirements then it is failure"

28) Difference between Priority and Severity

Parameters	Severity	Priority
Definition	Severity is absolute and Customer-Focused. 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	Priority is Relative and Business-Focused. 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.
Relation	Severity relates to the standards of quality.	Priority relates to the scheduling of defects to resolve them in software.
Value	The value of severity is objective.	The value of priority is subjective.
Change of Value	The value of Severity changes continually from time to time.	The value of Priority changes from time to time.
Who Decides the Defect	The testing engineer basically decides a defect's severity level	The product manager basically decides a defect's priority level.
Types	There are 5 types of Severities: Cosmetic, Minor, Moderate, Major, and Critical.	There are 4 types of Priorities: Critical, High, Medium, and Low.

29) What is Bug Life Cycle?



30) Explain the difference between Functional testing and NonFunctional testing?

SR NO	FUNCTIONAL TESTING	NON FUNTIONAL TESTING
1	Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements.	Non-Functional testingchecksthe Performance, reliability, scalability and other non-functional aspects of the software system.
2	Functional testing is executed first	Non functional testing should be performed after functional testing
3	Manual testing or automation tools can be used for functional testing.	Using tools will be effective for this testing.
4	Business requirements are the inputs to functional testing	Performance parameters like speed , scalability are inputs to non-functional testing
5	Functional testing describes what the product does	Nonfunctional testing describes how good the product works
6	Easy to do manual testing	Tough to do manual testing
7	Types of Functional testing are • Unit Testing • Smoke Testing • Sanity Testing • Integration Testing • White box testing • Black Box testing • User Acceptance testing • Regression Testing	Types of Nonfunctional testing are · • Performance Testing • Load Testing • Volume Testing • Stress Testing • Security Testing • Installation Testing • Penetration Testing • Compatibility Testing • Migration Testing

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

SR.NO	STLC (Software Testing Life Cycle)	SDLC (Software Development Life Cycle)
1.	Software Testing Life Cycle (STLC) is a process used to test software and ensure that quality standards are met.	A Software Development Life Cycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software.
2.	SDLC is mainly related to software development.	STLC is mainly related to software testing.
3.	Besides development other phases like testing is also included.	It focuses only on testing the software.
4.	SDLC involves total six phases or steps	STLC involves only five phases or steps
5.	In SDLC, more number of members (developers) are required for the whole process.	In STLC, less number of members (testers) are needed.
6.	In SDLC, development team makes the plans and designs based on the requirements.	In STLC, testing team(Test Lead or Test Architect) makes the plans and designs.
7.	Goal of SDLC is to complete successful development of software.	Goal of STLC is to complete successful testing of software.
8.	It helps in developing good quality software.	It helps in making the software defects free.
9.	SDLC phases are completed before the STLC phases.	STLC phases are performed after SDLC phases
10.	Post deployment support , enhancement , and update are to be included if necessary.	Regression tests are run by QA team to check deployed maintenance code and maintains test cases and automated scripts.

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

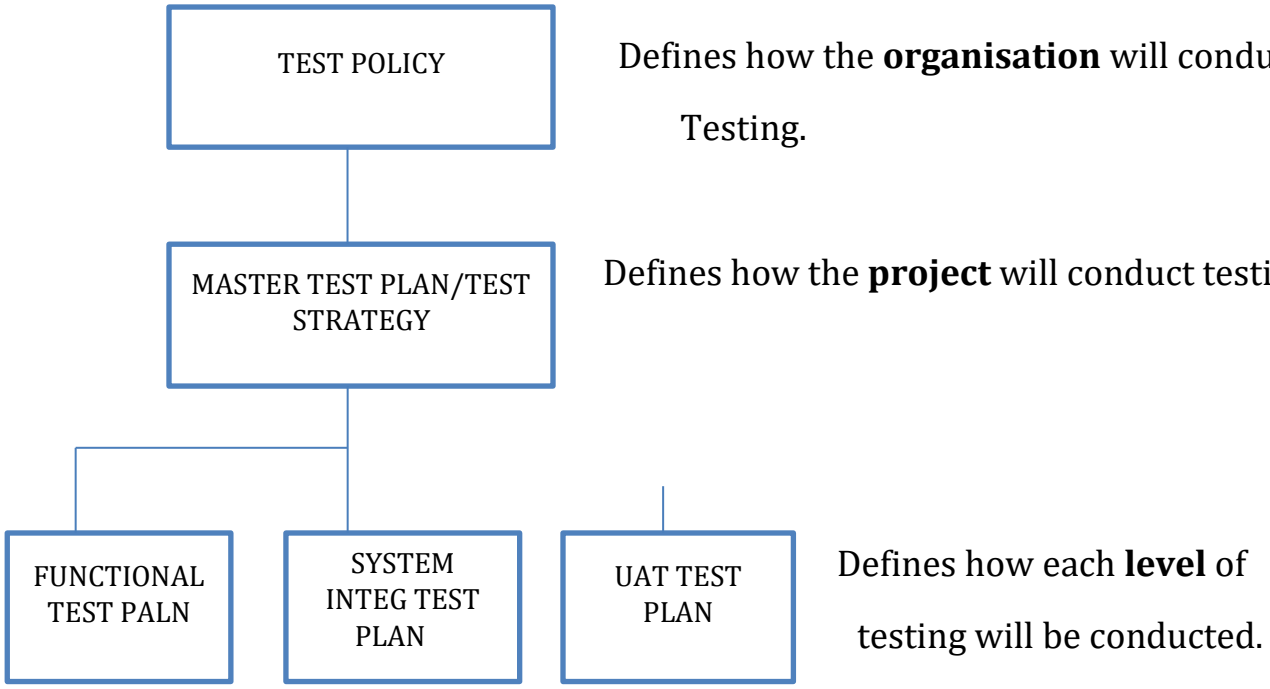
Sr no.	Test Scenarios	Test Case	Test Script
1.	Test Scenarios is any functionality that can be tested.	Test Case is a set of actions executed to verify particular features or functionality.	Test Script is a set of instructions to test an app automatically.
2.	Test Scenarios is derived from test artifacts like Business Requirement Specifaction (BRS) and Software Requirement Specification (SRS).	Test Case is mostly derived from test scenarios.	Test Script is mostly derived from test cases.
3.	Helps test end-to-end functionality in an agile way.	Helps in exhaustive testing of an app.	Helps to test specific things repeatedly.
4.	Test Scenarios is more focused on what to test.	Test Case is focused on what to test and how to test.	Test Script is focused on expected result.
5.	Test Scenarios takes less time and fewer resources to create.	Test Case requires more resources and time.	Test Script requires less time for testing but more resources for scripts creating and updating.
6.	Test Scenarios includes an end-to-end functionality to be tested.	Test Case includes test steps, data, expected results for testing.	Test Script includes different commands to develop a script.
7.	Test Scenarios main task is to check the full functionality of a software application.	Test Case main task is to verify compliance with the applicable standards, guidelines, and customer requirements.	Test Scripts main task is to verify that nothing is skipped, and the results are true as the desired testing plan.
8.	Test Scenario allows quickly assessing the testing scope.	Test Case allows detecting errors and defects.	Test Script allows carrying out an automatic execution of test cases.

33). Explain what Test Plan is? What is the information that should be covered.

A document describing the scope, approach, resources and schedule of intended test activities.
A Test Plan is a detailed document that catalogs the test strategies, objectives, schedule,

estimations, deadlines, and resources required to complete that project. All projects require a set of plans and strategies which define how the testing will be conducted.

There are number of levels at which these are defined:



32). What are the different Methodologies in Agile Development Model?

I. **Scrum:-** One of the most popular agile methodology examples is the agile scrum development methodology, which is depicted by various cycles of development. Similar to Kanban, Scrum breaks down the development phases into stages or cycles called 'sprints'. The development time for each sprint is maximized and dedicated, thereby managing only one sprint at a time. Scrum and agile methodologies focus on continuous deliverables, and thus this method lets designers adjust priorities to ensure that any incomplete or overdue sprints get more attention. Scrum Team has exclusive project roles such as a scrum master and a product owner with constant communications on the daily scrum where the activities are harmonized to devise the best way to implement the sprint.

II. **Kanban:**- Kanban is a very popular framework for development in the agile software development methodology. Originating from the Japanese language, the translation of the word 'Kanban' is "visual board or signboard" and is connected to the concept of "just in time"! Initially, the Kanban concept was introduced as a lean manufacturing system and slowly drove its way to agile software development teams. This method uses visual methods for developing and managing projects. Projects through Kanban are overseen with the help of the Kanban Board, which is divided into columns to depict the process flow of the software development. This helps in increasing the visibility of teams as the teams can see the progress through every stage of development and prepare for the upcoming tasks to deliver the product "just in time"! This method requires thorough interaction and transparency to enable the team members to be equipped with the right stage of development at any time and have a cohesive flow of work at all times.

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

	Authentication	Authorization
Purpose	Verifies User identity	Permits access to resources
Requirements	Identify credentials based on knowledge, possession, and inherence	Authenticated identity and access control policies
Responsibilities	Network security staff determine which factors to adopt. Users provide authentication factors when requesting access.	Leadership sets security strategies. Departments and workgroups define access criteria . Network security staff implement and maintain access control system.

Below problem faced in Web testing:-

- Cross Browser Compatibility
- Responsiveness
- Cross Device Compatibility
- Integration Testing
- Security
- Performance Testing

- Application getting slow
- Usability Testing
- Project deadline