1. What is SDLC?

SDLC or the Software Development Life Cycle is a process that produces software with the highest quality and lowest cost in the shortest time possible. SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use.

Why SDLC?

Here, are prime reasons why SDLC is important for developing a software system.

- It offers a basis for project planning, scheduling, and estimating
- Provides a framework for a standard set of activities and deliverables
- It is a mechanism for project tracking and control
- Increases visibility of project planning to all involved stakeholders of the development
- process
- Increased and enhance development speed
- Improved client relations
- Helps you to decrease project risk and project management plan overhead

2 SDLC Phases?

The entire SDLC process divided into the following stages:

- Phase 1: Requirement collection and analysis
- Phase 2: Feasibility study:
- Phase 3: Design:
- Phase 4: Coding:
- Phase 5: Testing:
- Phase 6: Installation/Deployment:
- Phase 7: Maintenance:

3.1 Requirements Gathering/Analysis.

This is a process with much communication taking place between stakeholders, end users and the project team. 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? These are general questions that get answered during a requirement gathering phase. The QA engineer playing the role to configure the requirements using requirements traceability matrix (RTM).

3.2 Design:

In this phase the software design is prepared from the requirement specifications which were studied in the first phase. System Design helps in specifying hardware and system requirements and also help in defining overall system architecture. In this phase the QA Engineers comes up with the Test strategy, where they mention what to test, how to test.

3.3 Implementation / Coding:

Upon receiving system design documents, the work is divided in modules/units and actual coding is started. Since, in this phase the code is produced so it is the main focus for the developer. This is the longest phase of SDLC. In this phase the QA Engineers comes up with the Test Environment setup and test Case Documentation.

3.4 Testing:

After the code is developed it is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. During this phase all types of like unit testing, integration testing, Smoke Testing, functional testing, Sanity Testing, system testing, acceptance testing is done as well as non-functional testing is also done.

3.5 Deployment: After successful testing the product is delivered / deployed to the customer for their use. As soon as the product is given to the customers, they will first do the beta testing/User

3 What are Popular SDLC models?

Waterfall model: Waterfall model works well for smaller projects where requirements are very well understood. The waterfall is a widely accepted SDLC model. In this approach, the whole process of the software development is divided into various phases. In this SDLC model, the outcome of one phase acts as the input for the next phase.

Incremental Model: The incremental model is not a separate model. It is essentially a series of waterfall cycles. The requirements are divided into groups at the start of the project. For each group, the SDLC model is followed to develop software. The SDLC process is repeated, with

each release adding more functionality until all requirements are met. In this method, every cycle act as the maintenance phase for the previous software release. Modification to the incremental model allows development cycles to overlap. After that subsequent cycle may begin before the previous cycle is complete.
V-Model: In this type of SDLC model testing and the development, the phase is planned in parallel. So, there are verification phases on the side and the validation phase on the other side. V-Model joins by Coding phase.
Agile Model: Agile methodology is a practice which promotes continue interaction of development and testing during the SDLC process of any project. In the Agile method, the entire project is divided into small incremental builds. All of these builds are provided in iterations, and each iteration lasts from one to three weeks. In 'Agile Model' after every sprint there is a demoable feature to the customer. Hence customer can see the features whether they are satisfying their need or not. Each release is thoroughly tested to ensure software quality is maintained. It is used for time critical applications.

Spiral Model: The spiral model is a risk-driven process model. This SDLC model helps the team to adopt elements of one or more process models like a waterfall, incremental, waterfall, etc. This model adopts the best features of the prototyping model and the waterfall model. The spiral methodology is a combination of rapid prototyping and concurrency in design and development activities.
Prototyping Model: Prototyping model is a software development model in which prototype is built, tested, and reworked until an acceptable prototype is achieved. It also creates base to produce the final system or software. It works best in scenarios where the project's requirements are not known in detail. It is an iterative, trial and error method which takes place between developer and client.

4 What is Software Testing?
According to ANSI/IEEE 1059 standard – A process of analyzing a software item to detect the differences between existing and required conditions (i.e., defects) and to evaluate the features of the software item.

5 What are the principles of software testing?
6.1 Testing shows presence of defects
Software Testing reduces the probability of undiscovered defects remaining in the software but even if no defects are found, it is not a proof of correctness.
6.2 Exhaustive testing is impossible
If you were testing this Operating system, you would realize that defects are likely to be found in multi-tasking activity and need to be tested thoroughly which brings us to our

next principal Defect Clustering.

6.3 Early testing

Early Testing - Testing should start as early as possible in the Software Development Life Cycle. So that any defects in the requirements or design phase are captured in early stages.

6.4 Defect clustering

Defect Clustering which states that a small number of modules contain most of the defects detected. If the same tests are repeated over and over again, eventually the same test cases will no longer find new bugs.

6.5 Pesticide Paradox

Repetitive use of the same pesticide mixes to eradicate insects during farming will over time led to the insect's developing resistance to the pesticide Thereby ineffective of pesticides on insects. The same applies to software testing. If the same set of repetitive tests are conducted, the method will be useless for discovering new defects. To overcome this, the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects.

6.6 Testing is context depending

Testing is context dependent which basically means that the way you test an ecommerce site will be different from the way you test a commercial off the shelf application. All the developed software's are not identical. You might use a different approach, methodologies, techniques, and types of testing depending upon the application type.

6.7 Absence of error - Fallacy

It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement. The absence of Error is a Fallacy i.e. Finding and fixing defects does not help if the system build is unusable and does not fulfill the user's needs & requirements.

6 What is STLC?

It is the testing process which is executed in systematic and planned manner. In STLC process, different activities are carried out to improve the quality of the product. Following steps are involved in Software Testing Life Cycle (STLC).

1. Requirement Analysis (RTM)
2. Test Planning (Test Strategy, Test Plan, Test Bed Creation)
3. Test Case Development (Test Procedures, Test Scenarios, Test Cases)
4. Environment Setup
5. Test Execution
6. Defect Reporting

7 **What are Quality Assurance and Quality Control?**

Quality Assurance:

Quality Assurance involves in process-oriented activities. It ensures the prevention of defects in the process used to make Software Application. So the defects don't arise when the Software Application is being developed. The process is:

- Plan - Organization should plan and establish the process related objectives and determine the processes that are required to deliver a high-Quality end product.
- Do - Development and testing of Processes and also "do" changes in the processes
- Check - Monitoring of processes, modify the processes, and check whether it meets the predetermined objectives
- Act - A Quality Assurance tester should implement actions that are necessary to achieve improvements in the processes.

Quality Control:

Quality Control involves in product-oriented activities. It executes the program or code to identify the defects in the Software Application.

8 What are the Quality assurance and Quality Control standards?

Quality assurance system standards, including ISO 9001, are defined as frameworks that provide regulations to organizations to ensure that their processes, inputs, products, and services are capable of meeting every customer requirement.

- Ensuring maximum satisfaction of clients by meeting their quality requirements
- Safety of products and services during usage
- Complying with international regulations and local legislative rules
- Being environmentally responsible
- Confidentiality of stakeholders including customers, employees, partners, and investors
- Assuring a safer workplace for employees
- Optimum allocation of resources and minimization of waste

9 What is Verification & Validation in software testing?

- Validation: Are we building the right system?
- Verification: Are we building the system, right?

In other words, validation is concerned with checking that the system will meet the customer's actual needs. validation is an extremely subjective process. Validation includes activities such as requirements modelling, prototyping and user evaluation. While verification is concerned with whether the system is well-engineered, error-free, and so on. Verification will help to determine whether the software is of high quality, but

it will not ensure that the system is useful. Verification includes all the activities associated with the producing high-quality software: testing, inspection, design analysis, specification analysis, and so on. It is a relatively objective process.

10 What is Test Plan Document?

Test plan document is a document which contains the plan for all the testing activities to be done to deliver a quality product. Test Plan document is derived from the Product Description, SRS, or Use Case documents for all future activities of the project. It is usually prepared by the Test Lead or Test Manager.

1. Test plan identifier
2. References
3. Introduction
4. Test items (functions)
5. Software risk issues
6. Features to be tested
7. Features not to be tested
8. Approach
9. Items pass/fail criteria
10. Suspension criteria and resolution requirements
11. Test deliverables
12. Remaining test tasks
13. Environmental needs
14. Staff and training needs
15. Responsibility
16. Schedule
17. Plan risks and contingencies
18. Approvals
19. Glossaries

11 What is Test Strategy?

Test Strategy is a high-level document (static document) and usually developed by project manager. It is a document which captures the approach on how we go about testing the product and achieve the goals. It is normally derived from the Business Requirement Specification (BRS). Documents like Test strategy doc is project-based document it can change according to project domain and requirements.

12 What is Test Suite?

Test Suite is a collection of test cases. The test cases which are intended to test an

application.

13 What is Test Scenario?

Test Scenario gives the idea of what we have to test. Test Scenario is like a high-level test case.

14 What is Test Case?

Test cases are the set of positive and negative executable steps of a test scenario which has a set of pre-conditions, test data, expected result, post-conditions and actual results.

15 What is Test Bed?

An environment configured for testing. Test bed consists of hardware, software, network configuration, an application under test, other related software.

16 What is Test Environment?

Test Environment is the combination of hardware and software on which Test Team performs testing. Example:

- Application Type: Web Application
- OS: Windows
- Web Server: IIS
- Web Page Design: Dot Net
- Client-Side Validation: JavaScript
- Server-Side Scripting: ASP Dot Net
- Database: MS SQL Server
- Browser: IE/Firefox/Chrome

17 What is Test Data?

Test data is the data that is used by the testers to run the test cases. Whilst running the test cases, testers need to enter some input data. To do so, testers prepare test data. It can be prepared manually and also by using tools.

18 What is Test Harness?

Test Harness in Software Testing is a collection of stubs, drivers and other supporting tools required to automate test execution. Test harness executes tests by using a test library and generates test reports. Test harness contains all the information needed to compile and run a test like test cases, target deployment port (TDP), source file under test, stubs, etc.

19 What is Test Closure?

Test Closure is the note prepared before test team formally completes the testing process. This note contains the total no. of test cases, total no. of test cases executed, total no. of defects found, total no. of defects fixed, total no. of bugs not fixed, total no of bugs rejected etc.,

20 What is Risk Factor and its Types?

In software testing Risks are the possible problems that might endanger the objectives of the project stakeholders. It is the possibility of a negative or undesirable outcome. A risk is something that has not happened yet and it may never happen; it is a potential problem. The types of Risk in a Test Project can be broadly categorized as

1. Strategy Risk: This includes Budget, Communication and Management risks
2. Project Definition Risks: This includes Project target, Scope, and requirements risks.
3. Human Resources Risk: This includes Skill, Team members and organization risks.

21 **What are the tasks of Test Closure activities in Software Testing?**

Test Closure activities fall into four major groups.

Test Completion Check: To ensure all tests should be either run or deliberately skipped and all known defects should be either fixed, deferred for a future release or accepted as a permanent restriction.

Test Artifacts handover: Tests and test environments should be handed over to those responsible for maintenance testing. Known defects accepted or deferred should be documented and communicated to those who will use and support the use of the system.

Lessons learned: Analyzing lessons learned to determine changes needed for future releases and projects. In retrospective meetings, plans are established to ensure that good practices can be repeated and poor practices are not repeated.

Result: Archiving results, logs, reports, and other documents and work products in the CMS (configuration management system).

22 List out Test Deliverables?

1. Test Strategy
2. Test Plan
3. Effort Estimation Report
4. Test Scenarios
5. Test Cases/Scripts
6. Test Data
7. Requirement Traceability Matrix (RTM)
8. Defect Report/Bug Report

9. Test Execution Report
10. Graphs and Metrics
11. Test summary report
12. Test incident report
13. Test closure report
14. Release Note
15. Installation/configuration guide
16. User guide
17. Test status report
18. Weekly status report (Project manager to client)

23 What is RTM?

Requirements Traceability Matrix (RTM) is used to trace the requirements to the tests that are needed to verify whether the requirements are fulfilled. Requirement Traceability Matrix AKA Traceability Matrix or Cross Reference Matrix.

Which Parameters to include in Requirement Traceability Matrix?

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

24 Types of Traceability Test Matrix

In Software Engineering, traceability matrix can be divided into three major components as mentioned below:

- Forward traceability: This matrix is used to check whether the project progresses in the
- forward desired direction and for the right product. It maps requirements to test cases.
- Backward or reverse traceability: It is used to ensure whether the current product
- remains on the right track. It maps test cases to requirements.
- Bi-directional traceability (Forward + Backward): This traceability matrix ensures that all
- requirements are covered by test cases. It analyzes the impact of a change in
- requirements affected by the Defect in a work product and vice versa.

25 How to create Requirement Traceability Matrix?

On the basis of the Business Requirement Document (BRD) and Technical Requirement Document (TRD), testers start writing test cases.

Step 1: Our sample Test Case is

"Verify Login, when correct ID and Password is entered, it should log in successfully"

Step 2: Identify the Technical Requirement that this test case is verifying. For our test case, the technical requirement is T94 is being verified.

Step 3: Note this Technical Requirement (T94) in the Test Case.
Step 4: Identify the Business Requirement for which this TR (Technical Requirement-T94) is defined
Step 5: Note the BR (Business Requirement) in Test Case
Step 6: Do above for all Test Cases. Later Extract the First 3 Columns from your Test Suite. RTM in testing is Ready!

26 Advantage of Requirement Traceability Matrix

- It confirms 100% test coverage
- It highlights any requirements missing or document inconsistencies
- It shows the overall defects or execution status with a focus on business requirements
- It helps in analyzing or estimating the impact on the QA team's work with respect to
- revisiting or re-working on the test cases.

27 What is Test Coverage?

Test Coverage states which requirements of the customers are to be verified when the testing phase starts. Test Coverage is a term that determines whether the test cases are written and executed to ensure to test the software application completely, in such a way that minimal or NIL defects are reported.

28 How to achieve Test Coverage?

The maximum Test Coverage can be achieved by establishing good 'Requirement Traceability'.

- Mapping all internal defects to the test cases designed
- Mapping all the Customer Reported Defects (CRD) to individual test cases for the
- future regression test suite.

29 Methods of Software Testing

There are 3 methods of software testing.
1) White box 2) Black box 3) Grey Box

30 What is White Box Testing?

White Box Testing is also called as Glass Box, Clear Box, and Structural Testing. It is based on applications internal code structure. In white-box testing, an internal perspective of the system, as well as programming skills, are used to design test cases. This testing usually was done at the unit level.

31 What is Black Box Testing?

Black Box Testing is a software testing method in which testers evaluate the functionality of the software under test without looking at the internal code structure. This can be applied to every level of software testing such as Unit, Integration, System and Acceptance Testing.

32 What is Grey Box Testing?

Grey box is the combination of both White Box and Black Box Testing. The tester who works on this type of testing needs to have access to design documents. This helps to create better test cases in this process.

33 What is Alpha Testing?

Alpha testing is done by the in-house developers (who developed the software) and testers. Sometimes alpha testing is done by the client or outsourcing team with the presence of developers or testers.

It has two phases:

- In the first phase of alpha testing, the software is tested by in-house developers. They
- use debugger software. The goal is to catch bugs quickly.
- In the second phase of alpha testing, the software is handed over to the software QA
- staff, for additional testing in an environment that is similar to the intended use.

34 What is Beta Testing?

Beta testing is done by a limited number of end users before delivery. Usually, it is done in the client places.

35 What is Gamma Testing?

Gamma testing is done when the software is ready for release with specified requirements. It is done at the client place. It is done directly by skipping all the in-house testing activities.

36 What is Functional Testing?

In simple words, what the system actually does is functional testing. To verify that each function of the software application behaves as specified in the requirement document. Testing all the functionalities by providing appropriate input to verify whether the actual output is matching the expected output or not. It falls within the scope of black box testing and the testers need not concern about the source code of the application.

37.1 What is Unit/Module Testing?

Unit Testing is also called as Module Testing or Component Testing. It is done to check

whether the individual unit or module of the source code is working properly. It is done by the developers in the developer's environment.

37.2 What is Integration Testing?

Integration Testing is the process of testing the interface between the two software units. Integration testing is done by three ways. Big Bang Approach, Top-Down Approach, Bottom-Up Approach.

37.3 What is System Testing?

Testing the fully integrated application to evaluate the system's compliance with its specified requirements is called System Testing End to End testing. Verifying the completed system to ensure that the application works as intended or not.
System testing is carried out by specialist testers or independent testers.
System testing should investigate both functional and non-functional requirements of the testing.

37.4 What is Smoke Testing?

Smoke Testing is done to make sure if the build we received from the development team is testable or not. It is also called as "Day 0" check. It is done at the "build level". It helps not to waste the testing time to simply testing the whole application when the key features don't work or the key bugs have not been fixed yet.

37.5 What is Sanity Testing?

Sanity Testing is done during the release phase to check for the main functionalities of the application without going deeper. It is also called as a subset of Regression testing. It is done at the "release level". We perform sanity testing when we don't have enough time for regression testing.

37.6 What is Regression Testing?

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 components.
Usually, we do regression testing in the following cases:

1. New functionalities are added to the application
2. Change Requirement (In organizations, we call it as CR)
3. Defect Fixing
4. Performance Issue Fix
5. Environment change (E.g., Updating the DB from MySQL to Oracle)

37.7 What is Retesting Testing?

Retesting is done to make sure that the tests cases which failed in last execution are passed after the defects are fixed. Retesting is carried out based on the defect fixes. In Retesting, the cases which are failed earlier can be included to check if the functionality failure in an earlier build.

37.8 What is Exploratory Testing?

Usually, this process will be carried out by domain experts. They perform testing just by exploring the functionalities of the application without having the knowledge of the requirements.

37 What is Monkey Testing?

Perform abnormal action on the application deliberately in order to verify the stability of the application.

38 What is Big Bang Approach?

Combining all the modules once and verifying the functionality after completion of individual module testing.

39 What is Top-Down Approach?

Testing takes place from top to bottom. High-level modules are tested first and then lowlevel modules and finally integrating the low-level modules to a high level to ensure the system is working as intended. Stubs are used as a temporary module if a module is not ready for integration testing.

40 What is Bottom-Up Approach?

It is a reciprocate of the Top-Down Approach. Testing takes place from bottom to up. Lowest level modules are tested first and then high-level modules and finally integrating the high-level modules to a low level to ensure the system is working as intended. Drivers are used as a temporary module for integration testing.

41 What is User Acceptance Testing / UAT? imp

It is also known as pre-production testing. This is done by the end users along with the testers to validate the functionality of the application. After successful acceptance testing. Formal testing conducted to determine whether an application is developed as per the requirement. It allows the customer to accept or reject the application. Types of acceptance testing is Alpha, Beta & Gamma.

42 What is Positive and Negative Testing?

Positive Testing: It is to determine what system supposed to do. It helps to check whether the application is justifying the requirements or not.

Negative Testing: It is to determine what system not supposed to do. It helps to find the defects from the software.

44. What is Non-Functional Testing?

In simple words, how well the system performs is non-functionality testing. Nonfunctional testing refers to various aspects of the software such as performance, load, stress, scalability, security, compatibility etc., Main focus is to improve the user experience on how fast the system responds to a request.

44.1. What is Performance Testing?

This type of testing determines or validates the speed, scalability, and/or stability characteristics of the system or application under test. Performance is concerned with achieving response times, throughput, and resource-utilization levels that meet the performance objectives for the project or product.

44.2. What is Load Testing?

It is to verify that the system/application can handle the expected number of transactions and to verify the system/application behavior under both normal and peak load conditions.

44.3. What is Volume Testing?

It is to verify that the system/application can handle a large amount of data.

44.4. What is Stress Testing?

It is to verify the behavior of the system once the load increases more than its design expectations.

44.5. What is Scalability Testing?

Scalability testing is a type of non-functional testing. It is to determine how the application under test scales with increasing workload.

44.6. What is Concurrency Testing?

Concurrency testing means accessing the application at the same time by multiple users to ensure the stability of the system. This is mainly used to identify deadlock issues.

44.7. What is GUI Testing?
Graphical User Interface Testing is to test the interface between the application and the end user.

44.8. What is Recovery Testing?
Recovery testing is performed in order to determine how quickly the system can recover after the system crash or hardware failure. It comes under the type of non-functional testing.

44.9. What is Installation Testing?
It is to check whether the application is successfully installed and it is working as expected after installation.

44.10. What is Formal Testing?
It is a process where the testers test the application by having pre-planned procedures and proper documentation.

44.11. What is Risk Based Testing?
Identify the modules or functionalities which are most likely cause failures and then testing those functionalities.

44.12. What is Compatibility Testing?
It is to deploy and check whether the application is working as expected in a different combination of environmental components.

44.13. What is Usability Testing?
To verify whether the application is user-friendly or not and was comfortably used by an end user or not. The main focus in this testing is to check whether the end user can understand and operate the application easily or not. An application should be selfexploratory and must not require training to operate it.

44.14. What is Security Testing?
Security testing is a process to determine whether the system protects data and maintains functionality as intended.

45 What is Soak Testing?
Running a system at high load for a prolonged period of time to identify the performance problems is called Soak or Endurance Testing.

46 What is Fuzz Testing?

Fuzz testing is used to identify coding errors and security loopholes in an application. By inputting massive amount of random data to the system in an attempt to make it crash to identify if anything breaks in the application.

47 What is Adhoc Testing?

Ad-hoc testing is quite opposite to the formal testing. It is an informal testing type. In Adhoc testing, testers randomly test the application without following any documents and test design techniques. This testing is primarily performed if the knowledge of testers in the application under test is very high. Testers randomly test the application without any test cases or any business requirement document.

48 What is Interface Testing?

Interface testing is performed to evaluate whether two intended modules pass data and communicate correctly to one another.

49 What is Reliability Testing?

Perform testing on the application continuously for long period of time in order to verify the stability of the application.

50 What is Bucket Testing? imp

Bucket or Split testing is a method to compare two versions of an application against each other to determine which one performs better.

51 What is Defect Cascading in Software Testing?

Defect cascading in Software testing means triggering of other defects in an application. When a defect is not identified or goes unnoticed while testing, it invokes other defects. It leads to multiple defects in the later stages and results in an increase in a number of defects in the application.

For example, if there is a defect in an accounting system related to negative taxation then the negative taxation defect affects the ledger which in turn affects other reports such as Balance Sheet, Profit & Loss etc.,

52 What is Walk Through?

A walkthrough is an informal meeting conducts to learn, gain understanding, and find defects. The author leads the meeting and clarifies the queries raised by the peers in the meeting.

53 What is Inspection?

Inspection is a formal meeting lead by a trained moderator, certainly not by the author. The document under inspection is prepared and checked thoroughly by the reviewers before the meeting. In the inspection meeting, the defects found are logged and shared with the author for appropriate actions. Post inspection, a formal follow-up process is used to ensure a timely and corrective action.

54 Who are all involved in an inspection meeting?

Author, Moderator, Reviewer(s), Scribe/Recorder and Manager.

55 What is a Defect?

The variation between the actual results and expected results is known as a defect. If a developer finds an issue and corrects it by himself in the development phase, then it's called a defect.

56 What is an Error?

We can't compile or run a program due to a coding mistake in a program. If a developer unable to successfully compile or run a program, then they call it as an error.

57 What is a Failure?

Once the product is deployed and customers find any issues then they call the product as a failure product. After release, if an end user finds an issue, then that particular issue is called as a failure.

58 What is Bug Severity?

Bug/Defect severity can be defined as the impact of the bug on customer's business. It can be Critical, Major or Minor. In simple words, how much effect will be there on the system because of a particular defect.

59 What is Bug Priority?

Defect priority can be defined as how soon the defect should be fixed. It gives the order in which a defect should be resolved. Developers decide which defect they should take up next based on the priority. It can be High, Medium or Low. Most of the times the priority status is set based on the customer requirement.

60 Tell some examples of Bug Severity and Bug Priority?

High Priority & High Severity: Submit button is not working on a login page and customers are unable to login to the application

Low Priority & High Severity: key feature failed but there's no impact on customer business, e.g., calculation fault in yearly report which end user won't use on daily basis.
High Priority & Low Severity: Spelling mistake of a company name on the homepage
Low Priority & Low Severity: FAQ page takes a long time to load

61 What is a Critical Bug?

A critical bug is a show stopper which means a large piece of functionality or major system component is completely broken and there is no workaround to move further. For example, Due to a bug in one module, we cannot test the other modules because that blocker bug has blocked other modules. Bugs which affect the customers' business are considered as critical.

Example:

1. "Sign In" button is not working on Gmail App and Gmail users are blocked to login to their accounts.
2. An error message pops up when a customer clicks on transfer money button in a Banking website.

62 What are entry criteria?

Entry criteria is a set of conditions that permits a task to perform, or in absence of any of these conditions, the task cannot be performed.

- The requirement document should be available.
- Complete understanding of the application flow is required.
- The Test Plan Document should be ready.

63 What is exit criteria?

Exit criteria is a set of expectations; this should be met before concluding the STLC phase.

- Test Cases should be written and reviewed.
- Test Data should be identified and ready.
- Test automation script should be ready if applicable.

64 What is the Test Management Reviews & Audit?

Management Review is also known as Software Quality Assurance or (SQA). It focuses more on the software process rather than the software work products. Quality Assurance is a set of activities designed to ensure that the project manager follows the standard process which is already pre-defined. In other words, Quality Assurance makes sure, the Test Manager is doing the right things in the right way.

An audit is the examination of the work products and related information to assesses whether the standard process was followed or not.

65 What is the difference between functional and non-functional testing?

Functional Testing	Non Functional Testing
Performed before non-functional testing	Performed after functional testing
Based on customer requirements	Based on customers' expectations
Describes what the product does	Describes how the product works

66 On what basis the acceptance plan is prepared?

Basically, the acceptance document is prepared using the following inputs.

- **Requirement document:** It specifies what exactly is needed in the project from the customers perspective.
- **Input from the customer:** This can be discussions, informal talks, emails, etc.
- **Project plan:** The project plan prepared by the project manager also serves as good input to finalize your acceptance test.

67 What is a Test Report? What does it include?

Test report is basically a document that includes a total summary of testing objectives, activities, and results. It is very much required to reflect testing results and gives an opportunity to estimate testing results quickly. It helps us to decide whether the product is ready for release or not. It also helps us determine the current status of the project and the quality of the product. A test report must include the following details:

- Test Objective
- Project Information
- Defect
- Test Summary