**MODULE 2 - MANUAL TESTING**

**Q.1 What is exploratory testing?**

**Ans:** Exploratory testing is an experience based testing in which no documentation or less documentation is needed and tester does testing with exploring the system.

**Exploratory Testing** is a type of testing in which the tester is free to select any possible methodology to test the software. It is an unscripted approach to software testing. In exploratory testing, software developers use their personal learning, knowledge, skills, and abilities to test the software developed by themselves. Exploratory testing checks the functionality and operations of the software as well as identify the functional and technical faults in it. The aim of exploratory testing is to optimize and improve the software in every possible way.

**Q.2 What is traceability matrix?**

**Ans**: Traceability matrix is a table type document that is used in the development of software application to trace requirements to every component.

It is prepared before the test execution process to make sure that every requirement is covered in the form of a Test case so that we don't miss out any testing.

It has three types,

Forward Traceability: Mapping of Requirements to Test cases

Backward Traceability: Mapping of Test Cases to Requirements

Bi-Directional Traceability: A Good Traceability matrix is the References from test cases to basis documentation.

It can be used both forward and backward tracing. And it is known as Requirement Traceability Matrix.

**Q.3 What is Boundary value testing?**

**Ans**: Boundary value analysis is a type of black box testing.

It is quite similar to equivalence partitioning, where we divide software into equal parts.

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 based on testing the boundary values of valid and invalid partitions. The behavior at the edge of the equivalence partition is more likely to be incorrect than the behavior within the partition, so boundaries are an area where testing is likely to find defects.

It checks for the input values near the boundary that have a higher chance of error. Every partition has its maximum and minimum values and these maximum and minimum values are the boundary values of a partition.

**Q.4 What is Equivalence partitioning testing?**

**Ans:** Equivalence partitioning is a black box type of testing.

In this testing, we have to divide test data into equivalent partitions and then test every partitions at least once.

Aim is to treat groups of inputs as equivalent and to select one representative input to test them all.

Equivalence partitioning is the process of defining the optimum number of tests by,

1-Reviewing documents such as the Functional Design Specification and Detailed Design Specification, and identifying each input condition within a function,

2- Selecting input data that is representative of all other data that would likely invoke the same process for that particular condition.

If the requirement is**a range of values**, then derive the test case for **one valid** and **two invalid**inputs.

Here, the**Range of values**implies that whenever we want to identify the range values, we go for equivalence partitioning to achieve the minimum test coverage. And after that, we go for error guessing to achieve maximum test coverage.

**Q.5 What is Integration testing?**

**Ans:** Integration testing is the testing in which two or more unit or component are combined together and tested that it works fine and does not have much defect.

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.

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.

Test drivers and test stubs are used in Integration Testing.

This test is performed by both developer and tester.

This test has two approaches,

**1 Big bang integration testing:** In this all components are combined together as group and then tested.

**2 Incremental integration testing:** In this one by one components are combined together and then tested.

It has two types;

**1: Component integration testing:** In this the components are combined together and then tested. Testing performed to expose defects in the interfaces and interaction between integrated components

**2: System integration testing:** In this the system is tested by connecting subsystems.

Sub systems are tested by connecting to the main system for finding if there are any defects.

**Q.6 What determines the level of risk?**

**Ans:** A factor that could result in future negative consequences, usually known as risk.

A Risk could be any future event with a negative consequence.

Determining the level of risk usually involves trying to assess not only the likelihood of an identified risk from actually occurring, but also the potential magnitude the consequences this risk could have on an organisation and its stakeholder, should it occur.

Once risks are identified through testing and analysis, they can then be assigned a ‘level of risk’ based on a set of pre-conceived criteria.

For example High, Medium and Low risks.

**Q.7 What is Alpha testing?**

**Ans:**  It is the form of Acceptance 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 always performed in Virtual Environment

It is a type of software testing performed to identify bugs before releasing the product to real users or to the public.

The purpose of alpha testing is to identify and resolve critical bugs and issues in the software before it is released to the public.

**Q.8 What is beta testing?**

**Ans:** Beta testing is performed by real time user or customers in real time environment.

It is always performed by the customers at their own site.

Beta Testing is always open to the market and public.

It is always performed outside the organization.

Testing performed by the client, stakeholder, and end-user.

Beta testing always is done after the alpha testing, and before releasing it into the market.

It is only a kind of Black Box Testing.

**Q.9 What is component testing?**

**Ans:** Component testing is a type of testing in which the units or components of a system or software are tested individually.

It is only tested by developers.

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

Unit testing frameworks, drivers, stubs and mock or fake objectsare used to assist in unit testing

Unit tests are typically written and run by software developersto ensure that code meets its design and requirements.

It is done when coding phase finishes.

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct.

Unit testing is performed by using the White Box Testing method.

**Q.10 What is functional system testing?**

**Ans**: Functional testing is based on an analysis of the specification of the functionality of a component or system.

It is a type of software testing which is used to verify the functionality of the software application, whether the function is working according to the requirement specification.

Functional testing ensures that the requirements or specifications are properly satisfied by the application

each function tested by giving the value, determining the output, and verifying the actual output with the expected value.

Functional testing performed as black-box testing which is presented to confirm that the functionality of an application or system behaves as we are expecting. It is done to verify the functionality of the application.

Each & every functionality of the system is tested by providing appropriate input, verifying the output and comparing the actual results with the expected results.

This testing involves checking of User Interface, APIs, Database, security, client/ server applications and functionality of the Application under Test. The testing can be done either manually or using automation.

**Q.11 What is Non-Functional Testing?**

**Ans:** Non-Functional Testing: Testing the attributes of a component or system that do not relate to functionality like reliability, efficiency, usability, interoperability, maintainability etc.

 It verifies whether the behavior of the system is as per the requirement or not.

It tests all the aspects which are not tested in functional testing.

The primary purpose of non-functional testing is to test the reading speed of the software system as per non-functional parameters.

Non functional testing is always done after functional testing finishes.

It is the testing of how the system works. Non-functional testing may be performed at all test levels.

To address this issue, performance testing is carried out to check & fine tune system response times.The goal of performance testing is to reduce response time to an acceptable level.

Non-functional testing is also very important as functional testing because it plays a crucial role in customer satisfaction.

For example, non-functional testing would be to test how many people can work simultaneously on any software.

**Q.12 What is GUI testing?**

**Ans:** GUI stands for Graphical User Interface.

GUI testing is the process for ensuring proper functionality of the graphical user interface for a specific application.

GUI testing generally evaluates a design of elements such as layout, colors and also fonts, font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links, and content.

It contains several visual elements, such as buttons, text boxes, menus, checkboxes, images, etc. GUI testing refers to the validating UI functions or features of an application that are visible to the users

GUI testing aims to ensure that the end-user gets a hassle-free experience. Since users are often unaware of the specific UIs, they focus on the design of the app, its colors, and whether it is easy to navigate.

You can perform GUI testing both manually and automatically. But automated is the most reliable solution in many cases.

**Q.13 What is Adhoc testing?**

**Ans:** Adhoc testing is an experience based testing. In which we test to break the system.

And this testing is just experience based and we just pick random clicking and we can say by just error guessing.

We just randomly guess error that is why it is also called error guessing testing.

It does not follow any kind of documentation and any kind of test cases.

It does not follow any test design techniques to create test cases.

This testing is primarily performed if the knowledge of testers in the system under test is very high.

Testers randomly test the application without any test cases or any business requirement document.

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.

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.

adhoc testing can be performed when there is limited time to do testing.

It has three types;

**1: Buddy testing**: In this one tester and one developer together do testing

**2: Pair testing**: In these two testers do testing together

**3: Monkey testing**: it is also like error guessing technique. We just try to find defect thru random clicking.

**Q.14 What is load testing?**

**Ans**: - It’s kind of a performance testing to check system behavior under load. It is a non functional testing.

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 together or back to back.

It is used to identify the total count of users that can access the application back to back.

**Q.15 What is stress Testing?**

**Ans:** It is a part of non functional 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 defined as a type of software testing that verifies the stability and reliability of the system.

This test particularly determines the system on its robustness and error handling under extremely heavy load conditions. It even tests beyond the normal operating point and analyses how the system works under extreme conditions.

Stress testing is performed to ensure that the system would not crash under crunch situations.

Stress testing is used to test the stability & reliability of the system

It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions.

stress testing is to determine the limit, at which the system or software or hardware breaks.

**Q.16 What is white box testing and list the types of white box testing?**

**Ans:** White box testing means that the tester knows the inside structure or we can say that the internal coding of the system.

It is mainly done by the developers. It is also called glass box testing

White box testing is a software testing technique that involves testing the internal structure and workings of a software application.

Testing based upon the structure of the code. Typically undertaken at Component and Component Integration Test phases by development teams

Developers do white box testing. In this, the developer will test every line of the code of the program. The developers perform the White-box testing and then send the application or the software to the testing team.

Test/code coverage is the percentage amount which shows the we have tested particular amount of items to total items.

It has three types,

**1: Statement or segment coverage:** it covers only true conditions. It is also known as segment and line coverage. In this technique, the aim is to traverse all statements at least once. Hence, each line of code is tested. Through statement coverage we can identify the statements executed and where the code is not executed because of blockage. In this process each and every line of code needs to be checked and executed.

**2: Decision or Branch coverage:** it covers both true and false conditions. Aim is to demonstrate that all Decisions have been run at least once. In this technique, test cases are designed so that each branch from all decision points is traversed at least once.

**3: Conditions coverage:** In this all conditions are must be covered. This is closely related to decision coverage but has better sensitivity to the control flow. full condition coverage does not guarantee full decision coverage. Condition coverage reports the true or false outcome of each condition

**Q.17 What is black box testing? What are the different black box testing techniques?**

**Ans:** Black box testing is type of testing in which the tester does not know the internal coding or structure but still they do testing based on inputs and expected outputs.The testers have no knowledge of how the system or component is structured inside the box.The technique of testing without having any knowledge of the interior workings of the application is Black Box testing.The tester is oblivious to the system architecture and does not have access to the source code.

There are few types of black box testing,

**1: Equivalence partitioning**: In this tester divides equivalent parts of the system to test.

which input data divided into partitions of valid and invalid values, and it is mandatory that all partitions must exhibit the same behavior. It is often seen that many types of inputs work similarly so instead of giving all of them separately we can group them and test only one input of each group.

**2: Boundary value analysis:** It is quite similar to EP because we do partitioning in this technique too. But difference is, in this we just check boundary values and just higher and lower values of boundary. Boundary Value Technique is used to test boundary values, boundary values are those that contain the upper and lower limit of a variable. It tests, while entering boundary value whether the software is producing correct output or not.

**3: Decision table:** It is used for combination of inputs. It is a table, if we put combination of inputs and possible outputs. Decision Table Technique is a systematic approach where various input combinations and their respective system behavior are captured in a tabular form. It is appropriate for the functions that have a logical relationship between two and more than two inputs.

**4: State transition testing:** It is a system where you get a different output for the same input.

State Transition Technique is used to capture the behavior of the software application when different input values are given to the same function. This applies to those types of applications that provide the specific number of attempts to access the application.

**Q.18 Mention what are the categories of defects?**

**Ans:** There are five types of defects,

**1: Database defect:** If there is any mistake in data, the system will not give expected output. It is called database defect.

e.g – 1) Values not deleted or inserted into the database properly,

2) Improper or wrong or null values inserted in place of the actual values

**2: Critical functionality defect:** The bug that affects main and crucial functionality of the system.

e.g not opening website when we paste URL.

**3: Functionality defect:** It is a defect when a bug affects functionality of the system.

e.g Coding errors, Buttons like Save, Delete, Cancel not performing their intended functions

**4: Security defect:** Application security defects generally involve improper handling of data sent from the user to the application.

e.g 1-Authentication: Accepting an invalid username or password

2- Authorization: Accessibility given to those who does not have permission.

**5: User interface defect:** : As the name suggests, the bugs deal with problems related to UI are usually considered less severe.

e.g Spelling mistakes, Alignment problems

**Q.19 Mention what bigbang testing is?**

**Ans:** It is a type of integration testing. In which we do testing with combining the unit or components altogether in a group.

 Big bang integration testing is a testing methodology in which all components or modules of a system are combined and tested as a whole. It is often used when it is not practical to test all components together incrementally.

**Q.20 What is the purpose of exit criteria?**

**Ans:** Exit criteria has a purpose of showing that when do we need to stop testing.

The Exit criteria define how much testing is enough and when testing activities can be declared complete.

The exit criteria get evaluated at the end of the testing cycle and is defined in Test Plan. It is the set of conditions or activities which must be fulfilled in order to conclude testing.

**Q.21 When should "Regression Testing" be performed?**

**Ans:** Regression testing is performed after any feature added, bug fix, or configuration changes.

**Q.22 What is 7 key principles? Explain in detail?**

**Ans:** The 7 key principles are,

**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.

**2: Exhaustive testing is impossible:**

it is very hard to test all the modules and their features.

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.

exhaustive testing of complex software applications requires enormous resources, it is too expensive and it takes too long. It is therefore impractical.

**3: Early testing:**

It shows that in testing it is needed to start testing as early as possible.

early testing means that all the testing activities should start in the early stages of the SDLC.

Early testing to identify the defects because if we find the bugs at an early stage, it will be fixed in the initial stage itself, which may cost us very less as compared to those which are identified in the future phase of the testing process.

**4: Defect clustering:**

The defect clustering defined that throughout the testing process, we can detect the numbers of bugs which are found in a small number of modules.

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.

**5: The pesticide paradox:**

is principle defined that if we are executing the same set of test cases again and again over a particular time, then these kinds of the test will not be able to find the new bugs in the software or the application. To get over these pesticide paradoxes, it is needed to create new test cases and apply it.

**6: Testing is context dependent:**

Testing is basically context dependent.

Testing is done differently in different contexts different kinds of sites are tested differently.

There is a definite way to test the commercial site as well as the e-commerce websites because every application has its own needs, features, and functionality.

To check this type of application, we will take the help of various kinds of testing, different technique, approaches, and multiple methods.

**7: Absence of error 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.

**Q:23 Difference between QA v/s QC v/s Tester.**

**Ans:**

**Tester:**

Activities which ensure the identification of bugs/error/defects in the Software.

Tester does all the test activities as per the requirements.

It is subset of QC.

It is product oriented process

It focuses on actual testing

Validating the product against specifications

It is a preventive process

**Quality Control:**

Activities which ensure the verification of developed software with respect to documented requirements.

Focuses on actual testing by executing Software with intend to identify bug/defect through implementation of procedures and process.

Product oriented activities.

It is a corrective process.

QC is subset of Quality Assurance

**Quality assurance:**

Focuses on processes and procedures rather than conducting actual testing on the system

Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements.

Process oriented activities.

Preventive activities

It is a subset of Software Test Life Cycle

**Q.24 Difference between Smoke and Sanity?**

**Ans:**

**Smoke testing:** It is done when initial builds of software are unstable.

verify that the critical functionalities of software are working fine.

It is executed before any detailed functional or regression tests are executed.

The main purpose of smoke testing is to reject a software application with defects so that QA team does not waste time testing broken software application.

**Sanity testing:**

It is performed after receiving a software build, with minor changes in code, or functionality.

 The initial aim of performing sanity testing is to determine that the planned features work roughly as expected.

If the sanity test fails, the build is rejected to save the costs and time complex in more severe testing.

**Q.25 Difference between verification and Validation.**

**Ans:**

**Verification:**

The verifying process includes checking documents, design, code, and program

Verification is the static testing.

It does *not* include the execution of the code

Methods used in verification are reviews, walkthroughs, inspections and desk-checking.

It can find the bugs in the early stage of the development.

Quality assurance team does verification.

It comes before validation

It consists of checking of documents/files and is performed by human.

Verification is for prevention of errors.

**Validation:**

It includes testing and validating the actual product.

Validation is the dynamic testing.

It includes the execution of the code.

Methods used in validation are Black Box Testing, White Box Testing and non-functional testing.

It can only find the bugs that could not be found by the verification process.

Validation is executed on software code with the help of testing team.

It comes after verification.

It consists of execution of program and is performed by computer.

Validation is for detection of errors.

**Q.26 Explain types of Performance testing.**

**Ans:** **Performance Testing** is a software testing process used for testing the speed, response time, stability, reliability, scalability, and resource usage of a software application under a particular workload.

Types:

**Load testing:**

 checks the application’s ability to perform under anticipated user loads. The objective is to identify performance bottlenecks before the software application goes live.

The load testing is used to check the performance of an application by applying some load which is either less than or equal to the desired load is known as load testing.

**Stress testing:**

involves testing an application under extreme workloads to see how it handles high traffic or data processing. The objective is to identify the breaking point of an application.

The stress testing is testing, which checks the behavior of an application by applying load greater than the desired load.

**Endurance testing:**

It is done to make sure the software can handle the expected load over a long period of time.

**Spike testing:**

 It is done to test the software’s reaction to sudden large spikes in the load generated by users.

**Volume testing:**

Under Volume Testing large number of data is populated in a database, and the overall software system’s behavior is monitored. The objective is to check software application’s performance under varying database volumes.

**Scalability testing:**

The objective of scalability testing is to determine the software application’s effectiveness in “scaling up” to support an increase in user load. It helps plan capacity addition to your software system.

**Q.27 What is Error, Defect, Bug and failure?**

**Ans.**

**Error:** any mistake in coding is error

**Defec**t: that error found by tester is defect

**Bug**: that defect accepted by developer is called bug

**Failure**: the system does not meet requirement and specification of client is a failure.

**Q.28 Difference between Priority and Severity.**

**Ans.**

**Priority:**

It is the order in which the developer should resolve a defect

Priority is associated with scheduling

Priority indicates how soon the bug should be fixed

Priority is driven by business value

Its value is subjective and can change over a period of time depending on the change in the project situation

High priority and low severity status indicates, defect have to be fixed on immediate bases but does not affect the application

Priority status is based on customer requirements

During UAT the development team fix defects based on priority

**Severity:**

Defect Severity is defined as the how much affect it does on system functionality.

Severity is associated with functionality or standards

Severity indicates the seriousness of the defect on the product functionality

Severity is driven by functionality

Its value is objective and less likely to change

High severity and low priority status indicates defect have to be fixed but not on immediate bases

Severity status is based on the technical aspect of the product

**Q.29 What is Bug Life Cycle?**

**Ans:**  Bug Life Cycle in software testing is the specific set of states that defect or bug goes through in its entire life.

It is also known as defect life cycle.

It goes phase to phase.

* **NEW:** When any new defect is identified by tester, it falls in ‘New’ state. It is first state of Bug Life Cycle.
* **Assigned:** Defects which are in status of ‘New’ will be approved and that newly identified defect is assigned to the development team for working on defect and to resolve that. When the defect is assigned to developer team then status of bug changes to ‘Assigned’ state.
* **Open:** in this Open state the defect is being addressed by developer team and developer team works on the defect for fixing the bug. Based on some specific reason if developer team feels that defect is not appropriate then it is transferred to either rejected or deferred state.
* **Fixed:** After necessary changes of codes or after fixing identified bug developer team marks state as fixed
* **Pending retest:** During the fixing of defect is completed, developer team passes new code to testing team for retest. And the code/application is pending for retesting at Tester side so status is assigned as Pending Retest
* **Retest:** At this stage, tester starts work of retesting defect to check whether defect is fixed by developer or not, and the status is marked as retesting
* **Reopen:** After retesting if tester team found that bug continues like previously even after developer team has fixed the bug, then status of bug is again changed to ‘Reopened’. Once again bug goes to ‘Open’ state and goes through life cycle again. This means it goes for Re-fixing by the developer team.
* **Verified:** The tester re-tests bug after it got fixed by developer team and if tester does not find any kind of defect/bug then bug is fixed and status assigned is verified
* **Closed:** It is the final state of Defect Cycle, after fixing defect by developer team when testing found that the bug has been resolved and it does not persist then they mark defect as a closed state
* **Rejected:** If the developer team rejects defect if they feel that defect is not considered as a genuine defect, and then they mark status as rejected
* **Not a defect:** f the defect has no impact or effect on other functions of the software then it is marked as not a defect
* **Deferred:**  If the developer team feels that defect that is identified is not a prime priority and it can get fixed in further updates or releases then developer team can mark status as deferred
* **Duplicate:** sometimes it may happen that defect is repeated twice or defect is same as any other defect then it is marked as duplicate.

**Q.30 Explain the difference between Functional testing and NonFunctional testing**

**Ans:**

**Functional testing:**

It verifies the operations and actions of an application.

It is based on requirements of customer

It helps to enhance the behavior of the application.

Functional testing is easy to execute manually.

It tests what the product does.

Functional testing is based on the business requirement.

It is performed before non-functional testing.

Helps to validate the behavior of the application.

**Non functional testing:**

It verifies the behavior of an application.

It is based on expectations of customer.

It helps to improve the performance of the application.

It is hard to execute non-functional testing manually.

It describes how the product does.

Non-functional testing is based on the performance requirement.

It is performed after the functional testing.

Helps to validate the performance of the application.

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

**Ans:**

**STLC:**

It stands for Software Testing Life Cycle

The only objective of the STLC phase is testing.

STLC is mainly related to software testing.

It focuses only on testing the software.

In STLC, less number of members are needed.

In STLC, testing team makes the plans and designs.

Goal of STLC is to complete successful testing of software

It helps in making the software defects free.

STLC phases are performed after SDLC phases.

**SDLC:**

It stands for Software Development Life Cycle

The main object of SDLC life cycle is to complete successful development of the software including testing and other phases.

SDLC is mainly related to software development

Besides development other phases like testing is also included.

In SDLC, more number of members are required for the whole process.

In SDLC, development team makes the plans and designs based on the requirements.

Goal of SDLC is to complete successful development of software.

It helps in developing good quality software.

SDLC phases are completed before the STLC phases.

**Q.32 What is the difference between test scenarios, test cases, and test script?**

**Ans:**

**Test Scenario**: It is defined as any functionality that can be tested. It is a collective set of test cases which helps the testing team to determine the positive and negative characteristics of the project.

A test scenario contains high-level documentation which describes an end to end functionality to be tested.

**Test case:** It is a set of actions executed to verify a particular feature or functionality of your software application. A test case contains test steps, test data, precondition, and postcondition developed for a specific test scenario to verify any requirement.

Test cases contain definite test steps, data, expected results for testing all the features of an application.

**Test script:** Test Script is set of instructions or a short program to test any functionality of software application.

Test Script is an automatic approach of software testing.

It is a program developed by the tester, intended to test any specific function of the software product.

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

**Ans:**  Test plan is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product. Test Plan helps us determine the effort needed to validate the quality of the application under test.

The information that should be covered,

1. Analyse the product
2. Design the Test Strategy
3. Define the Test Objectives
4. Define Test Criteria
5. Resource Planning
6. Plan Test Environment
7. Schedule & Estimation
8. Determine Test Deliverables

**Q.34 What is priority?**

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

Priority is defined as the order in which a defect should be fixed. Higher the priority the sooner the defect should be resolved.

**Low:** This defect can be fixed after the critical ones are fixed.

**Medium:**The defect should be resolved in the subsequent builds.

**High:** The defect must be resolved immediately because the defect is affecting the application to a considerable extent and the relevant modules cannot be used until it's fixed.

**Critical:** The defect must be resolved immediately because the defect is affecting the application or the product severely and the product cannot be used until it has been fixed.

**Q.35 What is severity?**

**Ans:** . It is the extent to which the defect can affect the software.

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

It is the classification of a defect based on its level of destructive impact on the requirements specification of the Software.

**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 or high**: 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 or moderate:** 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.

**Low or 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.

**Q.36 Bug categories are.**

**Ans:** bug categories are,

**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.

E.g. 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

**Q.37 Advantage of Bugzila.**

**Ans:**

Open source, free bug tracking tool.

Search option with advanced features.

File or Modify Bugs By Email.

Move Bugs Between Installs.

Multiple authentication Methods

Time Tracking.

Automated bug reporting; has an API to interact with system.

Integrated email capabilities.

Detailed permissions system.

Optimized database structure to enhance performance.

Robust security.

Powerful query tool.

Ideal for small projects.

**Q.38 What are the different Methodologies in Agile Development Model?**

**Ans: Scrum**: it is an agile development method which concentrates particularly on how to manage tasks within a team based development environment. Basically, Scrum is derived from activity that occurs during rugby match. Scrum believes in empowering the development team and advocates working in small teams, 7 to 9 members

There are three roles in it,

**Scrum Master:** The scrum can set up the master team, arrange the meeting and remove obstacles for the process

**Product owner:** The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.

**Scrum Team:** The team manages its work and organizes the work to complete the sprint or cycle.

**Sprint**: Sprint is a time-boxed period in which the scrum team needs to finish the set amount of work. Each sprint has a specified timeline of 1 weeks to 4 weeks

**Scrum roles:** Product owner, Scrum master, scrum team

**Artifacts of scrum:** Product backlog, sprint backlog, burndown chart

**Scrum ceremonies:** Sprint planning, daily scrum, sprint review

**Scrum Board:** Scrum Board is a board that shows the status of all the activities that need to be done within this sprint.

Scrum Board consists of four status,

**Open:** The 'Open' status means that the tasks which are available in 'Open' are not yet started.

**In progress:** The 'In progress' status means the developers completed their tasks.

**Testing**: The 'testing' means that the task is in a testing phase.

**Closed:** The 'closed' means the task has been completed.

**Q.39 Explain the difference between Authorization and Authentication in Web testing. What are the common problems faced in Web testing?**

**Ans:**

**Authorization**:

Authorization is a way to provide permission to someone to access a particular resource

Authorization determines what resources a user can access.

Authorization always takes place after authentication

Authorization isn’t visible to or changeable by the user.

While in this process, users or persons are validated.

**Authentication:**

Authentication is used to authenticate someone's identity

Authentication verifies who the user is

Authentication the identity of users are checked for providing the access to the system.

In the authentication process, users or persons are verified

It is done before the authorization process.

It needs usually the user’s login details.

**What are the common problems faced in Web testing?**

One of the main challenges of web application testing is ensuring that your web app works well across different browsers, devices, and operating systems.

Different browsers may have different rendering engines, standards support, features, and extensions that can affect how your web app looks and behaves.

**Q.40 When to use Usability Testing?**

**Ans:** **Usability Testing** also known as User Experience UX Testing, is a testing method for measuring how easy and user-friendly a software application is. A small set of target end-users, use software application to expose usability defects.

Usability testing mainly focuses on user’s ease of using application, flexibility of application to handle controls and ability of application to meet its objectives.

This testing is recommended during the initial design phase of SDLC, which gives more visibility on the expectations of the users.

**Q.41 What is the procedure for GUI Testing?**

**Ans: GUI Testing Techniques** can be categorized into three parts,

**Manual based approach:**Under this approach, graphical screens are checked manually by testers in conformance with the requirements stated in the business requirements document.

**Record and Replay:**GUI testing can be done using automation tools. This is done in 2 parts. During Record, test steps are captured by the automation tool. During playback, the recorded test steps are executed on the Application Under Test.

**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.

A model is a graphical description of a 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.

The following needs to be considered for this model based testing:

Build the model.

Determine Inputs for the model

Calculate the expected output for the model

Run the tests

Compare the actual output with the expected output

A decision on further action on the model.

**Q.42 HLR and TestCase of Instagram and facebook.**

**Ans: Instagram:**

<https://docs.google.com/spreadsheets/d/15IYzGhhd2ytyoT2568bxSfgDrxkKSGgeEKiZoBKKrfA/edit?usp=sharing>

**Facebook:**

<https://docs.google.com/spreadsheets/d/1_EpAkeNaJzDQ1yLKZx2F2p-sDxMis8XdkFWcBtxx-hw/edit?usp=sharing>

**Q.43 HLR and TestCase of Whatsapp web.**

**Ans:**

**Whatsapp web**

<https://docs.google.com/spreadsheets/d/1kEFsnZvv2hQM3LXcWDXjv-NUaY5DW2LFszSm-oAcgN8/edit?usp=sharing>

**Q.44 HLR and TestCase of Art of testing.**

**Ans:** Art Of Testing:

<https://docs.google.com/spreadsheets/d/1CgMcw57WWAG5w3RZPgv6bTDIonwsECMrlYsqFr6mGDU/edit?usp=sharing>

**Q.45 All scenarios.**

**Ans:**

<https://docs.google.com/spreadsheets/d/15BpBPW8Oe9qK8Cr2yQtDgIv967D63NYJzp98keNce-E/edit?usp=sharing>