**Assignment**

**Module -2 (manual testing)**

1. **What is Exploratory Testing?**

Exploratory testing is a concurrent process where

1. Test design, execution and logging happen simultaneously
2. Testing is often not recorded
3. Make use of experience, heuristics and test pattern
4. Testing is based on a test charter that may include

* Scope of the testing (in and out)
* The focused of exploratory testing is more on testing as a “thinking activity”
* A brief description of how test will be performed
* Is carried out in time boxed intervals
* Expected problem
* More structure than the error guessing

Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking.

Automation has its limit.

1. **What is traceability matrix?**

The condition should able to be linked back to their sources in the test basis this is known as traceability.

Types of traceability matrix

1. Forward traceability
2. Backward traceability
3. Bi-directional
4. **What is Boundary value testing?**

Boundary value analysis is methodology for designing test case that concentrate software testing effort on cases near the limit of valid range

Boundary value analysis is method which refine equivalence

Boundary value analysis generate test cases that highlight error better than equivalence partitioning

The trick is to concentrate software testing effort at the extreme end of equivalence classes

At those point when input value change from invalid error are most likely to occur.

1. **What is Equivalence partitioning testing?**

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

EP can be used for all level of testing

If we want to test the following IF statement

IF value is between 1 & 100 (inclusive)

Eg.

Value >= **1** & value =< **100 .**

EP says that by testing just one value we have tested the partition(typically a midpoint value is used)It assume that,

1. If one value find a bug, the other probably will too.
2. If one doesn’t find a bug, the other probably won’t either

In EP we must identifying valid equivalence partition & invalid partition equivalence partitions where applicable (typically in range test)

The valid partition is bounded by the value 1 & 100

Plus, there are 2 invalid partitions

1. **What is Integration testing?**

Integration testing performed to expose defect in the interface and in the interaction between integrated component or system.

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

Component may be code module, operating system, hardware and even complete system.

There are two level of integration system

1. Component integration testing
2. System integration testing

Integration testing method

There are two type of method of integration testing

1. Big bang integration testing
2. Incremental integration testing
3. Top down approach
4. Bottom up approach
5. **Big bang integration testing**

In big bang integration testing all component or module 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 & difficult to trace the cause of failure because of the this late integration.

**Advantage**

Convenient for small system

**Disadvantage**

Fault localization is difficult.

Since the integration testing can commence only after “ALL” the module are designed, testing teamwill have less time for execution in the testing phase.

1. **Incremental integration testing**
2. **Top down approach**

Testing take place from top to bottom following the control flow or architectural structure

Eg. (Starting from the GUI or main menu ) component or system are substituted by stub.

In top down approach, testing take place from top to bottom following the control flow of the software system

Takes help of **stub** foe testing

**Advantage**

Fault localization is easier possibility to obtain an early prototype critical module are tested priority

**Disadvantage**

Need many **studs**

Module its lower level are tested inadequately

1. **Bottom up approach**

Testing take place from the bottom of the control flow upward component or system are substituted by driver.

In the bottom up strategy, each module at lower level is tested with higher module until all module are tested

It takes help of **driver** for testing

**Advantage**

Fault localization is easier, no time is wasted waiting foe all module to be developed unlike big bang approach.

**Disadvantage**

Critical module (at the top level of software architecture) which control the flow of application are tested last and may be prone defects.

Early prototype is not possible.

**Stubs & Driver**

Stubs and Driver are the dummy program in integration testing used to facilitate the software testing activity.

**Stub** is called by the module under test.

**Driver** called the module to be tested.

1. **What determines the level of risk?**

**Risk: -** A factor that could result in future negative consequence

**Type of risk**

There are two type of risk

1. Project risk
2. Product risk

**Type of example**

Example of project risk is senior team member leaving the project abruptly

Example of product risk would be flight reservation system not installing in the environment

Mitigation in this case would be conducting a smoke and sanity testing.

Accordingly, you will make change in your scope item to include sanity testing.

1. **What is component testing?**

**Component (Unit) testing: -** A minimal software item that can be tested in isolation. It means “ A until is the smallest testable part of software”.

Unit testing is a level of the software testing process where individual unit/component of a software /system are tested. The purpose is to validate that each unit of the software perform as designed.

Unit testing is the first level of testing and is performed prior to integration testing sometime known as unit testing module testing or program testing.

Unit testing framework, drive, stub & mock or fake object are used to assist in unit testing

Functional and Non-functional testing. Unit test are typically written and run by software developer to ensure that code meet its design to behaves as intended with debugging too.

Unit test find problem early in the development cycle.

Unit testing is performed by the using the white box testing method.

1. **What is functional system testing?**

Testing based on an analysis of the specification of the functionality of a component or system.

Specification: - e.g -Requirement specification, use case, functional specification or may be undocumented.

Functional: - What the system does

Functional test: - based on function and future may be applied at all test

e.g – component test, system test

consider the external (not internal) behavior of the software – Black box testing. What is does rather than how it does it more on this later. Each & every functionality of the system is tested by providing appropriate input, verifying the output & comparing the actual result with expected result.

This testing involves checking of user interface APIs, database, security, client/server application & functionality of the application under test. The testing can be done either manually or using automation.

1. **What is Non-Functional Testing?**

Non-functional testing checks the performance reliability, scalability and other non-functional aspects of the software system.

Non-functional testing should be performed after functional testing.

Using tools will be effective for this testing.

Performance parameter like speed, scalability are input to non-functional testing.

Non-functional testing describes how good the product works.

Tough to do manual testing.

**Type of Non-functional testing are**

Performance testing

Load testing

Volume testing

Stress testing

Security testing

Installation testing

Penetration testing

Compatibility testing

Migration testing

1. **What is Ad-hoc testing?**

Ad-hoc (**Error Guessing**) testing in an informal testing type with an aim to break the system.

In fact, it does not create test case altogether. This testing is primarily performed if the knowledge of tester in the system under test is very high.

Ad-hoc testing does not follow any structure way of testing & it is randomly done on any part of application. Main aim of this testing is to find defect by random checking.

Ad-hoc testing can be achieved with the testing technique called error guessing.

The error guessing can be done by the people having enough experience on the system to guess the most likely sources of error.

Using experience of postulate error. Use error guessing to complement test design technique.

There are three type of Ad-hoc testing

1. Buddy testing
2. Pair testing
3. Monkey testing
4. **What is white box testing and list the types of white box testing?**

White box testing based on an analysis of the internal structure of the component or system.

Structured based testing technique is also known as white box testing or glass box testing, technique because here the tester require knowledge of how the software is implanted how it worked.

**Test/code coverage**

Test coverage measures the amount of testing performed by a set of tests, wherever we can count things and can tell whether or not each of those things has been tested by some test, then we can measure coverage and is known as test coverage.

**Type of coverage**

1. **Statement coverage**

The statement coverage is also known as line coverage or segment coverage.

The statement coverage covers only the true conditions.

1. **Decision coverage**

Decision coverage also known as branch coverage or all edge coverage

It covers both the true and false condition unlikely the statement coverage.

Aim to demonstrate that all decision has run at least once.

1. **Condition coverage**

This is closely related to decision coverage but has better sensitivity to the control flow. However full condition coverage does not guarantee full decision coverage.

Condition coverage reports the true or false outcome of each condition.

Condition coverage measures the conditions independently of each other.

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

Black box testing – testing, ither 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 technique because they view the software as a black box with input and output.

The tester is oblivious to the system architecture and does not have access to the source code.

Typically, when performing a black box test, a tester will interact with the system users interface by providing input and examining output without knowing how and where the input are worked upon.

INPUT OUTPUT

**----------------→ ----------------→**

If output = Expected result then pass

**Type of black box testing**

* Equivalence partitioning
* Boundary value analysis
* Decision table
* State transition testing
* Use case
* Other black box testing

Syntax or pattern testing

1. **Equivalence partitioning**

Aim to treat group of input as equivalent and select one representative input to test them all.

EP can be used for all level 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.

If we want to test the following **IF statement: ”**If value is between 1 to 100 (inclusive) (E.g value >=1 and value <=100) **Then…..”**

We could put the a range of number .

1. **Boundary value analysis (B.V.A)**

Boundary value analysis is a methodology for designing test cases that concentrate software testing effort on cases near the limit of valid range.

Boundary value analysis is a method which refine equivalence partitioning.

Boundary value analysis generate test cases that highlight error batter than equivalence partitioning.

The trick is to concentrate software testing effort at the extreme ends of the equivalence classes.

At those point when input value change from valid to invalid error are most likely to occur.

Boundary value analysis uses the same analysis of partitioning as EP and is usually used in conjunction with EP in test cases design.

1. **Decision table**

The other two specification based software testing technique, decision table and state transition testing are more focused on business logic or business rules.

A decision table is good way to deal with combination of things.

This technique is sometimes also referred to as a ‘cause-effect’ table. The reason for this is that there is an associated logic diagramming technique called “cause effect graphing”.

Table based technique where

* Input to the system are recorded
* Output to the system are defined

Input are usually defined in terms of action which are Boolean (true and false)

Output are recorded against each unique combination of inputs.

Using the decision table the relationship between the input and the possible outputs are mapped together.

1. **State transaction testing**

**State transaction –** A transaction between two state of as component or system.

a black box test design technique in which test cases are designed to execute valid and invalid transaction. Also known as N-switch testing.

An excellent tool to capture certain type of system requirement and to document internal system design. As such can be used for a number of test level.

1. **What is Alpha testing?**

* It is always performed by the developers at the software development site.
* Sometimes it is also performed by the 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.
* Alpha testing is definitely performed and carried out at the developing organization location with the involvement of developers.
* It comes under the category of both white and black box testing.

1. **What is beta testing?**

* It is always performed by the customer 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.
* Beta testing is performed and carried out by user or you can say people at their own locations and site using customer data.
* It is only a kind of black box testing
* Beta testing is always performed at the time when software product and project are marketed
* It is always performed at the user premises in the absence of the development team.
* It is also consider as the user acceptance testing (UAT) which is done at customer or user area.
* Beta testing can be consider “pre-release” testing
* Pilot testing is testing to product on real world as well as collect data on the user of product in the classroom.

1. **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 control like menus, buttons, icons, and all type of bars -tool bar, menu bar, dialog boxes and windows etc.

**What do you check in GUI testing**

1. Check all the GUI element for size, position, width, length and acceptance of character or numbers. For instance, you must be able to provide input to the input fields.
2. Check you can execute the intended functionality of the application using the GUI
3. Check error messages are displayed correctly
4. Check for clear demarcation of different sections on screen
5. Check font used in application is reachable
6. Check the alignment of the text is proper
7. Check the color of the font and warning messages is aesthetically pleasing
8. Check that the images have good clarity
9. Check that the images are properly aligned
10. Check the positioning of GUI element for different screen resolutions
11. **What is load testing?**

* Load testing is to test the system behavior under normal workload conditions and it is just testing or simultaneously with the actual workload.
* Load testing identifies the bottleneck breaking the system under various workload and check the how the system reacts when the load is gradually increased.
* Load testing does not break the system.

1. **What is stress Testing?**

* Stress testing is to test the system behavior under extreme condition and is carried out till the system failure.
* Stress testing determine the point of the system to reveal the maximum point after which it break
* Stress testing tries to break the system by testing with over whekming data or resources.

1. **Mention what are the categories of defects?**

Defect can be categorized in to different types basing on the core issue they address.

1. **Data quality/ Database defect:**

* Deal with improper handling of data in the database
* Value not deleted / inserted in to the database properly
* Improper/ wrong/ null values inserted in place of the actual values

1. **Critical functionality defect:**

* The occurrence of these bug hampers the critical functionality of the application
* E.g : Exceptions

1. **Functionality defect:**

* These defects affect the functionality of the applications.

Examples:

* All JAVA script error
* Button like save, delete, cancel not performing their intended functions
* A missing functionality (or) a feature not functioning the way it is intended to continuous executions of loops.

1. Security defect:

* Application security defect generally involve improper handling of data sent from the user to the applications. These defects are the most severe and given highest priority for fix.
* **Authentications:**

Accepting an invalid username / password.

* **Authorizations:**

Accessibility to pages though permission not given.

1. **User interface defect:**

* As the name suggest, the bug deals with problem related to UI are usually considered less severe.

Examples

* Improper error/ warning/ UI message
* Spelling mistake
* Alignment problem

1. **Mention what big bang testing is?**

**Big bang integration testing**

In big bang integration testing all component or module 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 & difficult to trace the cause of failure because of this late integration.

**Advantage**

Convenient for small system

**Disadvantage**

Fault localization is difficult.

Since the integration testing can commence only after “ALL” the module are designed, testing team will have less time for execution in the testing phase.

1. **What is the purpose of exit criteria?**

* How do we know when to stop testing?
* Run out of time?
* Run out of budget
* The business tells you it went live last night?
* Boss says stop?
* All defect have been fixed?
* When out exit criteria have been met?
* Purpose of exit criteria is to define when to STOP testing either at the
* End of all testing

i.e product go live

* End of phase of testing

E.g Hand over from system test to UAT.

1. **When should "Regression Testing" be performed?**

Testing of a previously tested program following modification to ensure that defect have not been introduced or uncovered in unchanged area of the software as a result of the change made. It is performed when the software or its environment is changed.

If the test is re-run and passed you cannot necessary say the fault has been resolved because, you can also need to ensure that the modification have not caused unintended side effect elsewhere and that the modified system still meet its requirement regression testing.

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

General testing principle (7 key principle)

1. **Testing shows presence of defect:**

Testing can show that defect are present but can not prove that there are no defect.

1. **Exhaustive testing is impossible:**

Testing everythings including all combinations of input and preconditions is not possible

So, instead of doing the exhaustive testing we can use risk and priotise to focus testing efforts.

1. **Early testing:**

Testing activities should start as a early as possible in the software or system development life cycle and should be focused on defined objective.

1. **Defect clustering:**

A small number of module contain most of the defect discovered during pre-release testing or are possible for the most operational failures.

1. **Pesticide paradox:**

If the some test are repeated over and over again, eventually the same set of test case 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 test need to be written to exercise different parts of the software or system to potentially find more defects.

1. **Testing is context dependent:**

The testing is basically context dependent different kind of sites are tested differently

Safety:

Critical software is tested differently from an e-commerce sites.

1. **Absence of error fallacy:**

If the system built is unusable and does not fulfill the user needs and expectations then finding and fixing defect does not help.

1. **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 |

1. **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  the "stability" of the system in order to  proceed with more rigorous testing | The objective of the testing is to verify  the "rationality" of the system in order  to 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 |

1. **Difference between verification and Validation**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Verification** | **Validation** |
| Definition | 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 |
| Question | Are we building the product right? | Are we building the right product? |
| Evaluation Items | Plans, Requirement Specs, Design Specs, Code, Test Cases | The actual product/software |
| Activities | Reviews   Walkthroughs   Inspections | Testing |

1. **Difference between Priority and Severity**

|  |  |  |
| --- | --- | --- |
| **SR No.** | **Defect Severity** | **Defect Priority** |
| 1 | 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. Priority defines the order in which we should resolve a defect. Should we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mentioned then the developer has to fix it at the earliest. The priority status is set based on the customer requirements. |
| 2 | For example: If an application or web page crashes when a remote link is clicked, in  this case clicking the remote link by an user is rare but the impact of application  crashing is severe. So the severity is high but priority is low. | For example: If the company name is misspelled in the home page of the website,  then the priority is high and severity is low to fix it. |

1. **Explain types of Performance testing.**

Software performance testing is a mean of quality assurance (QA). It involves testing software application to ensure they will perform well under their expected workload.

Checking a software program.

* Speed:

Determine whether the application respond quickly.

* Scalability:

Determine the maximum user load the software application can handle.

* Stability:

Determine if the application is stable under varying load.

* Performance testing type

1. Load testing

It is 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 load to determine at what point the system response time degrades or fails.

1. Stress testing

Stress testing done to make sure that the system would not crash under crunch situation.

It even test beyond the normal operating point & evaluate how the system work under those extreme conditions.

Stress testing is also known as endurance testing.

1. Endurance testing

Type of testing which checks for memory leaks or other problems that may occur with prolonged execution. It is usually performed by performance engineers.

1. Volume testing

Testing which confirms that any values that may become large over time (such as accumulated counts, logs, and data files), can be accommodated by the program and will not cause the program to stop working or degrade its operation in any manner. It is usually conducted by the performance engineer.

1. Scalability testing

Part of the battery of non-functional tests which tests a software application for measuring its capability to scale up - be it the user load supported, the number of transactions, the data volume etc. It is conducted by the performance engineer.

1. Spike testing
2. **Explain the difference between Functional testing and Non-Functional testing**

|  |  |  |
| --- | --- | --- |
| **SR No.** | **Functional Testing** | **Non-Functional Testing** |
| 1 | Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements. | Non-Functional testing checks the  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 |

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

* Error:

A discrepancy between a computed, observed or measured value or conditions and the true specified or theoretically correct value or condition.

This can be misunderstanding of the internal state of the software an oversight in term of memory management, confusion about the proper way to calculate a value etc.

* Defect:

Commonly refer to several trouble with the software product with its external behavior or with its it internal features.

* Bug:

A fault in a program which cause the program to perform in an unintended or unanticipated manner.

Bug is a terminology of tester.

* Failure:

The inability of a system or component to perform its required functions within specified performance requirement.

E.g – bug crash, exception and fault.

1. **What is Bug Life Cycle?**

A computer bug is an error, flaw, mistake, failures or its in a computer program that prevent it from working correctly or produces an incorrect result.

Bug arises from mistake and error made by people, in either a programs sources code or its design.

* New:

When a new defect is logged and posted for the time. It is assigned a status as new.

* Assigned:

Once the bug is posted by the tester, the lead of the tester approve the bug and assign the bug to the developers team.

* Open:

The developer stars analyzing and work on the defect fix.

* Fixed:

When developers make a necessary code change and verifies the changes, her or she can make bug status as “ fixed ”.

* Pending retest:

Once the defect is fixed the developers gives a particular code for retesting the code to the tester. Since software testing remains pending from the tester end, the status assigned is “pending retest”.

* Retest:

Tester does the retesting od the code at this stage to check whether the defect is fixed by the developer or not and changes the status or “retest”.

* Verified:

The tester retest the bug after it got fixed by the developer.

If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified”.

* Reopen:

If the bug persist even after the developer has fixed the bug, the tester change the status to “reopened”.

Once again the bug goes through the life cycle.

* Closed:

If the bug is no longer exist then tester assign the status “closed”.

* Duplicates:

If the defect is repeated twice or the defect correspond to the same concept of the bug.

The status is change to “duplicates”.

* Rejected:

If the developer feel the defect is not a genuine defect then it changes the defect to “rejected”.

* Deferred:

If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status “deferred” is assigned to such bug.

* Not a bug:

If is does not affected the functionality of the application then the status assigned to a bug is “not a bug”.

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

| SDLC | STLC |
| --- | --- |
| SDLC is mainly related to software development. | STLC is mainly related to software testing. |
| Besides development other phases like testing is also included. | It focuses only on testing the software. |
| SDLC involves total six phases or steps. | STLC involves only five phases or steps. |
| In SDLC, more number of members (developers) are required for the whole process. | In STLC, less number of members (testers) are needed. |
| 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. |
| Goal of SDLC is to complete successful development of software. | Goal of STLC is to complete successful testing of software. |
| It helps in developing good quality software. | It helps in making the software defects free. |
| SDLC phases are completed before the STLC phases. | STLC phases are performed after SDLC phases. |
| 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. |
| Creation of reusable software systems is the end result of SDLC. | A tested software system is the end result of STLC. |

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

**Test cases**

A test case is a high-level document with instructions on the specific functionality of the software product to be tested.

A test case is the software development life cycle’s ‘What to test’ component.

Test cases are written in simple English.

A test case is a document with instructions on testing the specific functionality of an application.

Test scenarios serve as an outline for writing test cases

Test cases are primarily used in Manual Testing.

It ensures end-to-end test coverage with assumed data types.

Test cases take a lot of time and resources to document.

**Test scripts**

Test Script is a step-by-step instruction to test each software product’s functionality (test case)

Test script is the software development life cycle’s ‘How to test’ component.

Test scripts are written in programming languages like VB Script, Python, Java, etc.

Test Script is a program that runs various test data on the functionality of an application.

Test Script is a program that runs various test data on the functionality of an application.

Test Case serves as an outline for writing test scripts.

 Test scripts are widely used in Automation

It can serve as a reusable component. A single script can run tests with various types of data.

The time to execute Test scripts is less than the time to write test cases.

**Test scenarios**

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

It focuses on more “what to test” **than** “how to test”.

Test scenarios are a one-liner. So, there is always the possibility of ambiguity during the testing.

Test scenarios are derived from test artefacts like BRS, SRS, etc.

It helps in an agile way of testing the end to end functionality

Test scenarios are high-level actions.

Comparatively less time and resources are required for creating & testing using scenarios.

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

Test plan is determine as a document describing the scope , approach , resources and schedule.

Test plan should include following things

**1. Test Strategy and Objectives**Identify the main purpose of testing (in light of the product requirements) and what a successful completion of a testing cycle looks like. The two important factors in terms of quality assurance are usually coverage and velocity. The test strategy will define which one comes first (usually based on the business objectives).

**2. Schedule, Estimation, and Deliverables**  
This depends whether your test cycle occurs during or after the cycle, but it is important to be able to determine how long will it take to qualify the tested functionality. And no less important is to understand “what will I get” as a user by the end of the testing cycle — is there any traceability matrix report I can rely on in order to support the decision making process? Predictability is key in this area, as the business would always want to know when we should be ready to ship.

**3. Resources Required to Perform Testing**  
Given different coverage needs, it is important to understand what will it take — for example — to complete a testing cycle within x amount of days/hours. Resources include people (testing team) and technology (testing platforms/product units, testing tools — frameworks and other orchestration/execution tools).

1. **What is priority?**

Priority defines the order in which we should resolve a defect.

Example: if the company name is misspelled in the home page of the website, then the priority is high and severity is low fix it.

1. **What is severity?**

It is the extent to which the defect can affect the software. In other words it define the impact that a given defect has on the system.

1. **Bug categories are**

**Bugs categories are mention below**

1. Database defect
2. Critical defect
3. Functionality defect
4. Security defect
5. User interface defect

**Database defect**

Deal with improper handling of the data in the database .

Value not inserted in to database properly

Improper values inserted in place of the actual value

**Critical defect**

The occurrence of the bugs hampers the crucial functionality of the application

Exa exceptions

**Functionality defects**

These defects affects the functionality of the application

Exa java script

**Security defects**

Application security defect genrally involve import handling of data sent from the user to the improper handling of data sent from the user to the application .

These defects are the most severe given highest priority for a fix .

* Authentication
* **Authorisation**

User interface defects

As the name suggest the bugs deals with problem related to ui are usally considered less severe

e.x improper error

spelling mistake

alignment problem

1. **Advantage of Bugzila**

**The Advantages of Bugzilla are:**

* it is an open-source widely used bug tracker;
* it is easy in usage and its user interface is understandable for people without technical knowledge;
* it easily integrates withtest management instruments;
* it integrates with an e-mailing system;
* it automates documentation.

1. **Difference between priority and severity**

## Difference Between Severity and Priority in Testing

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Severity in Testing** | **Priority in Testing** |
| Definition | Severity is a term that denotes how severely a defect can affect the functionality of the software. | Priority is a term that defines how fast we need to fix a defect. |
| Parameter | Severity is basically a parameter that denotes the total impact of a given defect on any software. | Priority is basically a parameter that decides the order in which we should fix the defects. |
| 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 3 types of Priorities: High, Medium, and Low. |

1. **What are the different Methodologies in Agile Development**

**Model?**

Agile methodology is a “step by step” dynamic focused on short-term visibility but never losing the long-term product goal.

## Main Agile methodologies:

### **1. Scrum**

### Scrum is the most used of the many frameworks underpinning Agile methodology.

**Scrum is characterised by cycles or stages of development, known as sprints**

It is usually used in the management of the development of software products but can be used successfully in a business-related context.

### **2. Kanban**

The word Kanban is of Japanese origin and its meaning is linked to the concept of “just in time”.

the Kanban method is organised on a board or table (Kanban board), divided into columns, showing every flow within the software production project.

As the development evolves, the information contained in the table changes, and whenever a new task comes into play, a new “card” is created.

This methodology is also useful in individual business departments, such as HR, marketing, etc., bringing the desired visibility over all the team’s tasks.

The Kanban method **requires communication and transparency** so that the members of any team all know exactly what stage development is at and can see the status of a project at any time.

### **3. Extreme Programming**

This is a typical Agile development framework, developed by Kent Beck, and can be adapted to development companies of various dimensions.

**Extreme Programming**  methodology is based around the idea of discovering “the simplest thing that will work” without putting too much weight on the long-term product view.

Teamwork is extremely important in XP, since, when there is a problem, it is solved by the whole team of managers, developers or customers, bringing them together to promote conversation and engagement and break down barriers to communication.

### **4. Lean Development**

Lean development is a methodology that comes directly from Lean Manufacturing, created by Toyota, and applied to software development.

This method offers a conceptual framework and follows values, principles and good development practices that can be applied to an Agile development approach.

**Lean development forces the team to ruthlessly remove any activity that does not bring ultimate value to the product.**

### **5. Crystal**

This is a family of Agile methodologies, and **Crystal is one of the most flexible frameworks, giving tremendous freedom to the team to develop their own processes.**

It focuses way more on individuals and how they interact rather than on the process or the tools so communication is an essential key aspect.

Crystal has variants such as **Crystal Clear** (up to an 8-person team), **Crystal Yellow**(up to a 10 to 20-person team), **Crystal Orange** (up to a 20 to 50-person team) and **Crystal Red**(for big teams with 50 to 1000 people).

Crystal focuses on principles such as People, Interactions, Community, Skills, Talent and Communication, aiming to deliver the best possible software development process.

It’s a light methodology in terms of documentation, where teams can find their own ways over preferred work modalities, removing management overheads and creating a “free” process.

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

**Authentication :- a**ccepting an invalid user name / password

**Authorization :- a**ccessibility to pages through permission not given

In web testing some common problems are mentioned below

1. interoperability
2. security
3. performance
4. responsiveness
5. **When to used Usablity Testing?**

usability testing can and should be conducted on the current iteration of a product before beginning any new design work, after you’ve begun the strategy work around a brand new site or app.

This will quickly identify areas for opportunity, and reduce the amount of assumptions your design team will make with regard to what the user wants.

Additionally, after the usability tests analysis, the team should have the ability to pinpoint the steps needed  to achieve the project goals with as little disruption as possible.

1. **What is the procedure for GUI Testing?**

GUI is a software testing type that checks the graphical user interface of the software

* Check all the GUI element size position , width , length etc
* Check for error massages are displayed correctly
* Check for cleardemaraction of different section on screen
* Check font used in application is readable
* Check the alignment of text is proper
* Check the colour of the font and warning messages is aesthetically pleasing
* Check all the images have good celerity
* check all the images are properly aligned
* Check the positioning of GUI elements for different screen resolution