Module-2 Manual Testing Assignment

• **What is Exploratory Testing?**

- Exploratory testing is a concurrent process where,
    - Test design, execution and logging happen simultaneously
    - Makes use of experience, heuristics and test patterns
    - Carried out in time boxed intervals

• **What is traceability matrix?**

- Traceability matrix is a table type document that is used in the development of software application to trace requirements.
- It can be used for both forward (from Requirements to Design or Coding) and backward (from Coding to Requirements) tracing.
- 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.

• **What is Boundary value testing?**

- Boundary value testing is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges.

• **What is Equivalence partitioning 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.

• **What is Integration testing?**

- Integration Testing is a level of the software testing process where individual units are combined and tested as a group.
- There are 2 levels of Integration Testing as mentioned below,
  Component Integration Testing & System Integration Testing

• **What determines the level of risk?**

- In software testing scenario, the risk level is determined by two dimensions: probability and impact.
- **Probability**: It measures the likelihood of an event occurring, typically expressed as a percentage or qualitative scale. It answers the question: "How likely is it to happen?" It ranges from 0 to 100%. The probability can never be 0% because it indicates that there is no risk, and it can never be 100% as it indicates that it is no longer a risk but a certainty.

- **Impact**: Risk, by default, brings a negative impact to any project. It assesses the consequences or severity of an event, usually quantified in monetary, temporal, or qualitative terms. It answers the question: “What would be the extent of its effect?”

- By taking into account these two factors, you can calculate the level of the risk: Level of Risk in Software = Probability of Risk Occurring X Impact of the occurred risk

• **What is Alpha testing?**

- 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’s location with the involvement of developers.

- It comes under the category of both White Box Testing and Black Box Testing.

• **What is beta testing?**

- Beta testing is always open to the market and public.

- It is conducted for the 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 (field testing) is performed and carried out by users or you can say people at their own locations and site using customer data.

- It is only a kind of Black Box Testing.

- It is also considered as the User Acceptance Testing (UAT) which is done at customers or users area.

• **What is component testing?**

- Component testing is the first level of testing and is performed prior to Integration Testing.

- Sometimes known as Unit Testing, Module Testing or Program Testing

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

- Component Testing is the testing of individual software components.

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

- **What is functional system testing?**

- Functional System Testing is a requirement that specifies a function that a system or system component must perform.

- Types of Functional testing are,

➢ Unit Testing

➢ Smoke Testing

➢ Sanity Testing

➢ Integration Testing

➢ White box testing

➢ Black Box testing

➢ User Acceptance testing

➢ Regression Testing

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

- Non-Functional Testing is a testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portability.

- Types of Nonfunctional testing are,

➢ Performance Testing

➢ Load Testing

➢ Volume Testing

➢ Stress Testing

➢ Security Testing

➢ Installation Testing

➢ Penetration Testing

➢ Compatibility Testing

➢ Migration Testing

- **What is GUI Testing?**

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

- **What is ADHOC testing?**

- The ADHOC (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.

- **What is load testing?**

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

- **What is stress Testing?**

- Stress testing system is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load.

- **What is white box testing and list the types of white box testing?**

- White Box Testing is testing based on an analysis of the internal structure of the component or system.

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

- In white-box testing the tester is concentrating on how the software does it. For example, a structural technique may be concerned with exercising loops in the software.

- The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

- **Types of White-box Testing**
  - ➢ Experience Based Testing
  - ➢ Grey-box Testing
  - ➢ Adhoc Testing
  - ➢ Exploratory Testing

- **What is black box testing? What are the different black box testing techniques?**

- Black-box testing: Testing, either functional or non-functional, without reference to the internal structure of the component or system.

- Specification-based testing technique is also known as 'black-box' or input/output driven testing techniques because they view the software as a black-box with inputs and outputs.

- The testers have no knowledge of how the system or component is structured inside the box.

- In black-box testing the tester is concentrating on what the software does, not how it does it.

- **Black-box Testing Technique**
  - ➢ Equivalence Partitioning (E.P.)
  - ➢ Boundary Value Analysis (B.V.A.)
  - ➢ Decision Table
  - ➢ State Transaction Testing

- **Mention what are the categories of defects?**

- Following are some of the basic categories of defects in the software development:

- **Arithmetic Defects:** It include the defects made by the developer in some arithmetic expression or mistake in finding solution of such arithmetic expression. This type of defects are basically made by the programmer due to access work or less knowledge. Code congestion may also lead to the arithmetic defects as programmer is unable to properly watch the written code.
- **Logical Defects:** Logical defects are mistakes done regarding the implementation of the code. When the programmer doesn't understand the problem clearly or thinks in a wrong way then such types of defects happen. Also while implementing the code if the programmer doesn't take care of the corner cases then logical defects happen. It is basically related to the core of the software.
- **Syntax Defects:** Syntax defects means mistake in the writing style of the code. It also focuses on the small mistake made by developer while writing the code. Often the developers do the syntax defects as there might be some small symbols escaped. For example, while writing a code in C++ there is possibility that a semicolon(;) is escaped.
- **Multithreading Defects:** Multithreading means running or executing the multiple tasks at the same time. Hence in multithreading process there is possibility of the complex debugging. In multithreading processes sometimes there is condition of the deadlock and the starvation is created that may lead to system's failure.
- **Interface Defects:** Interface defects means the defects in the interaction of the software and the users. The system may suffer different kinds of the interface testing in the forms of the complicated interface, unclear interface or the platform based interface.
- **Performance Defects:** Performance defects are the defects when the system or the software application is unable to meet the desired and the expected results. When the system or the software application doesn't fulfill the users's requirements then that is the performance defects. It also includes the response of the system with the varying load on the system.
- **Boundary and Range Defects:** These defects relate to issues where the software does not handle inputs or data outside of specified boundaries or ranges correctly. For example, not handling edge cases or boundary values appropriately.
- **Data Validation Defects:** It involves failing to validate user inputs, leading to potential security vulnerabilities or incorrect data handling.
- **Deployment Defects:** It includes setup mistakes, installation issues and problems that stop the software from functioning properly in a real-world setting.
- **Integration Defects:** When different software modules or components do not function well together, integration errors result. Problems with data synchronization, communication, and functionality may result from this.
- **Documentation Defects:** These refer to errors or issues in the accompanying documentation, including user manuals, help files or API documentation. Incomplete or inaccurate documentation can lead to user confusion.
- **Data Defects:** These can result in data corruption or loss, inconsistent data storage or issues with data retrieval. For example, data validation errors could lead to incorrect data being stored.

- **Mention what is Big Bang testing?**

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

- **What is the purpose of exit criteria?**

- Purpose of exit criteria is to define when we 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)

- **When should "Regression Testing" be performed?**

- Regression testing should be carried out:
  - when the system is stable and the system or the environment changes
  - when testing bug-fix releases as part of the maintenance phase

- **What is 7 key principles? Explain in detail?**

1. Testing shows presence of Defects
- Testing can show that defects are present, but cannot prove that there are no defects.
- Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.

2. Exhaustive Testing is Impossible!
- Testing everything including all combinations of inputs and preconditions is not possible.
- So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts.
- That is we must Prioritise our testing effort using a Risk Based Approach.

3. Early Testing
- Testing activities should start as early as possible in the software or system development life cycle, an should be focused on defined objectives.
- Testing activities should start as early as possible in the development life cycle.

4. Defect Clustering
- A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures.
- Defects are not evenly spread in a system.
- They are ‘clustered’

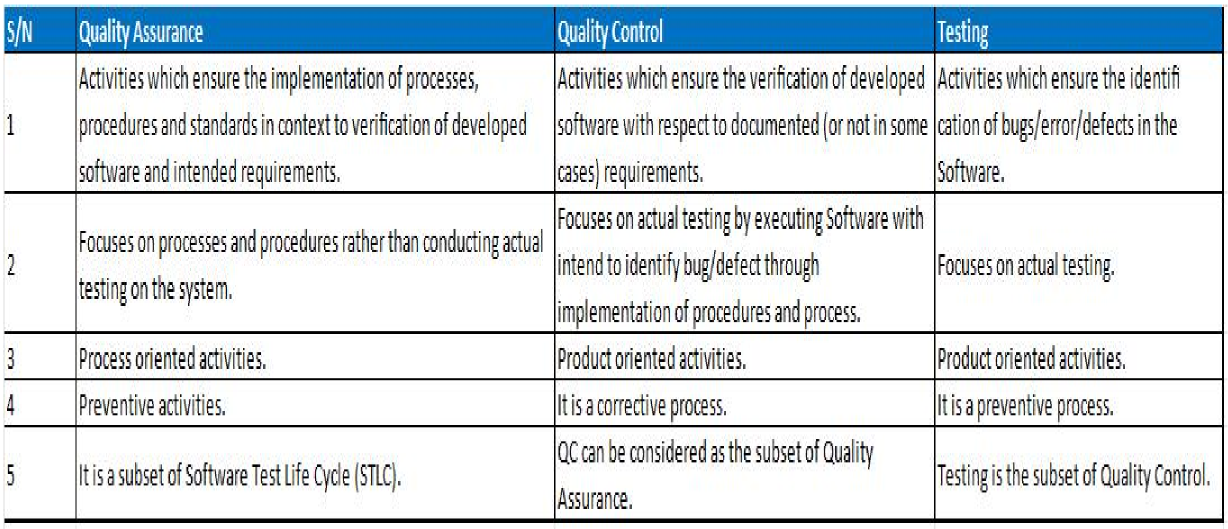
5. The Pesticide Paradox
- Testing identifies bugs, and programmers respond to fix them.
- As bugs are eliminated by the programmers, the software improves.
- As software improves the effectiveness of previous tests erodes.

6. Testing is Context Dependent
- Testing is basically context dependent.
- Testing is done differently in different contexts
- All so different industries impose different testing standards

7. Absence of Errors Fallacy

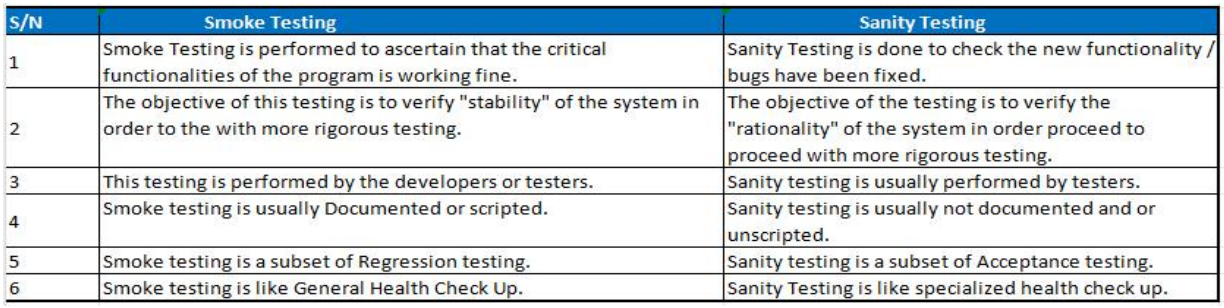
- If the system built is unusable and does not full fill the user’s needs and expectations then finding and fixing defects Does not help.
- Even after defects have been resolved it may still be unusable and/or does not fulfil the users’ needs and expectations.

- **Difference between QA v/s QC v/s Tester**



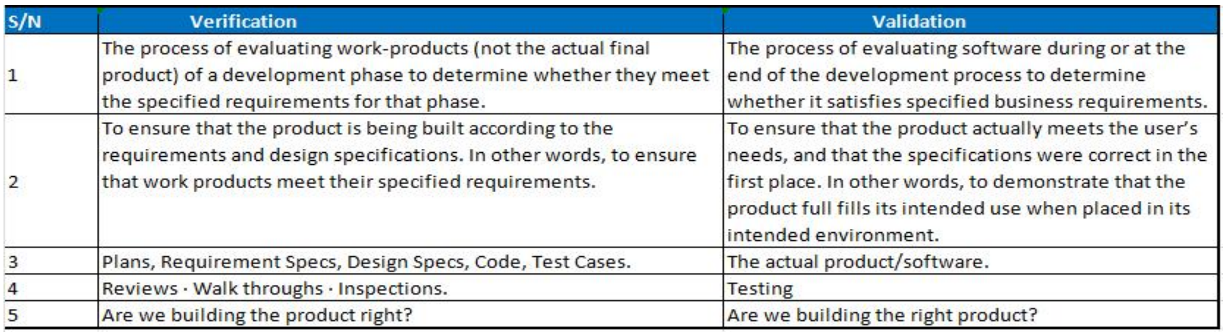
S/N	Quality Assurance	Quality Control	Testing
1	Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements.	Activities which ensure the verification of developed software with respect to documented (or not in some cases) requirements.	Activities which ensure the identifi cation 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.

- **Difference between Smoke and Sanity?**



S/N	Smoke Testing	Sanity Testing
1	Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine.	Sanity Testing is done to check the new functionality / bugs have been fixed.
2	The objective of this testing is to verify "stability" of the system in order to the with more rigorous testing.	The objective of the testing is to verify the "rationality" of the system in order proceed 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 or unscripted.
5	Smoke testing is a subset of Regression testing.	Sanity testing is a subset of Acceptance testing.
6	Smoke testing is like General Health Check Up.	Sanity Testing is like specialized health check up.

- **Difference between verification and Validation**



S/N	Verification	Validation
1	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.
2	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 full fills its intended use when placed in its intended environment.
3	Plans, Requirement Specs, Design Specs, Code, Test Cases.	The actual product/software.
4	Reviews · Walk throughs · Inspections.	Testing
5	Are we building the product right?	Are we building the right product?

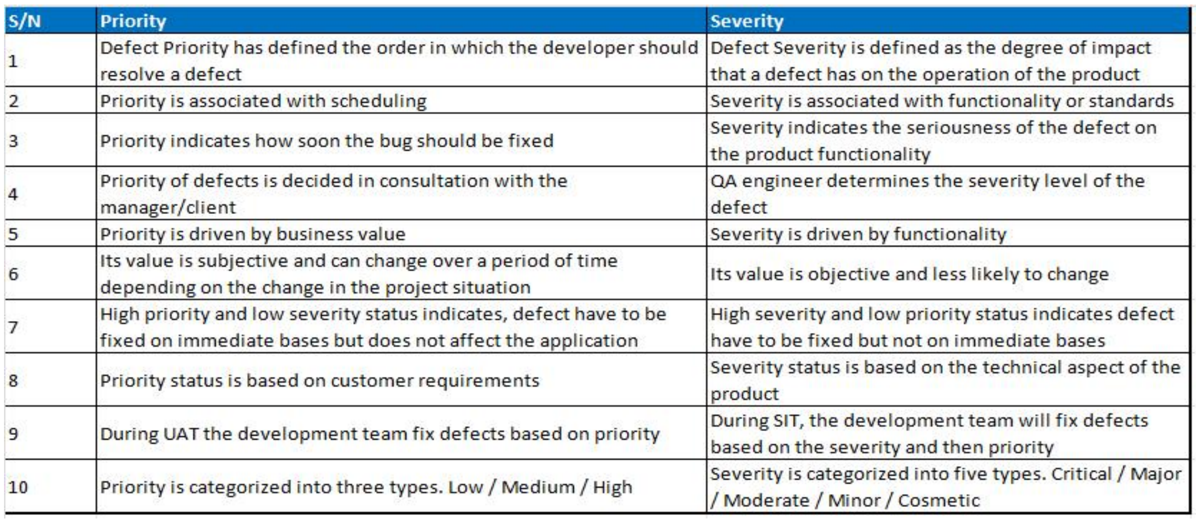
- **Explain types of Performance testing.**

- Load testing
- Stress testing
- Endurance testing
- Spike testing
- Volume testing
- Scalability testing

- **What is Error, Defect, Bug and failure?**

- **Error:** A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. This can be a misunderstanding of the internal state of the software, an oversight in terms of memory management, confusion about the proper way to calculate a value, etc.
- **Defect:** Commonly refers to several troubles with the software products, with its external behaviour or with its internal features.
- **Bug:** A fault in a program which causes the program to perform in an unintended or unanticipated manner. See: anomaly, defect, error, exception, and fault. Bug is terminology of Tester.
- **Fault:** An incorrect step, process, or data definition in a computer program which causes the program to perform in an unintended or unanticipated manner. See: bug, defect, error, exception.

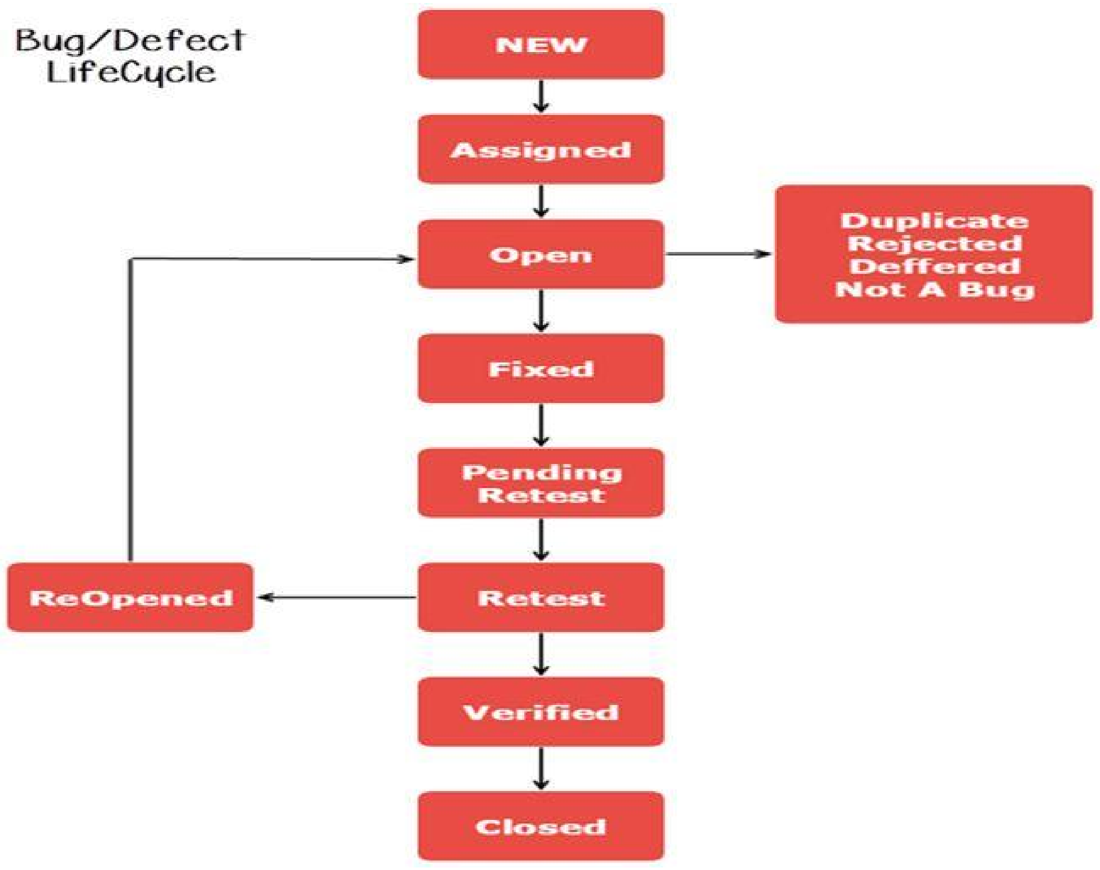
- **Difference between Priority and Severity**



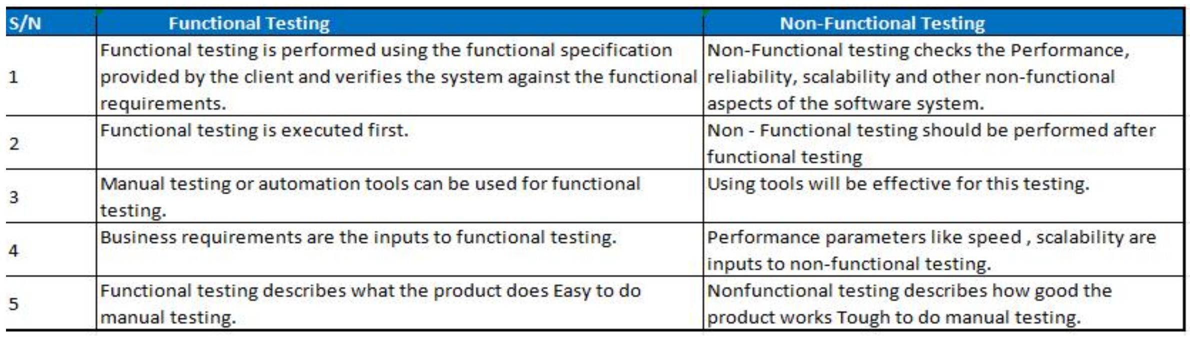
S/N	Priority	Severity
1	Defect Priority has defined the order in which the developer should resolve a defect	Defect Severity is defined as the degree of impact that a defect has on the operation of the product
2	Priority is associated with scheduling	Severity is associated with functionality or standards
3	Priority indicates how soon the bug should be fixed	Severity indicates the seriousness of the defect on the product functionality
4	Priority of defects is decided in consultation with the manager/client	QA engineer determines the severity level of the defect
5	Priority is driven by business value	Severity is driven by functionality
6	Its value is subjective and can change over a period of time depending on the change in the project situation	Its value is objective and less likely to change
7	High priority and low severity status indicates, defect have to be fixed on immediate bases but does not affect the application	High severity and low priority status indicates defect have to be fixed but not on immediate bases
8	Priority status is based on customer requirements	Severity status is based on the technical aspect of the product
9	During UAT the development team fix defects based on priority	During SIT, the development team will fix defects based on the severity and then priority
10	Priority is categorized into three types. Low / Medium / High	Severity is categorized into five types. Critical / Major / Moderate / Minor / Cosmetic

- **What is Bug Life Cycle?**

- A bug life cycle in software testing is a set of statuses designed to coordinate defect management. A bug status helps keep all the members of the development team posted on the progress. The cycle starts when a QA engineer reports a new issue found in the tested software and finishes when this issue is solved.
- The duration or time span between the first time defects is found and the time that it is closed successfully, rejected, postponed or deferred is called as ‘Bug Life Cycle’.



- **Explain the difference between Functional testing and Non-Functional testing**

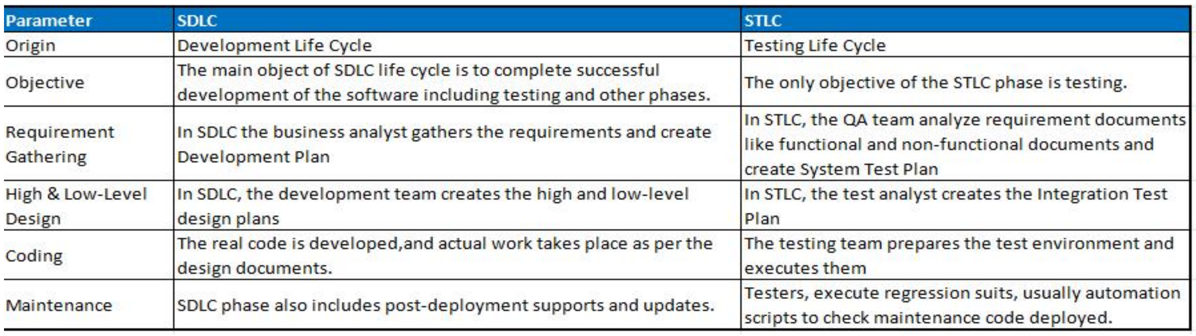


S/N	Functional Testing	Non-Functional Testing
1	Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements.	Non-Functional testing checks the Performance, reliability, scalability and other non-functional aspects 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 Easy to do manual testing.	Nonfunctional testing describes how good the product works Tough to do manual testing.

- **To create HLR & Test Case of Instagram and Facebook login page**

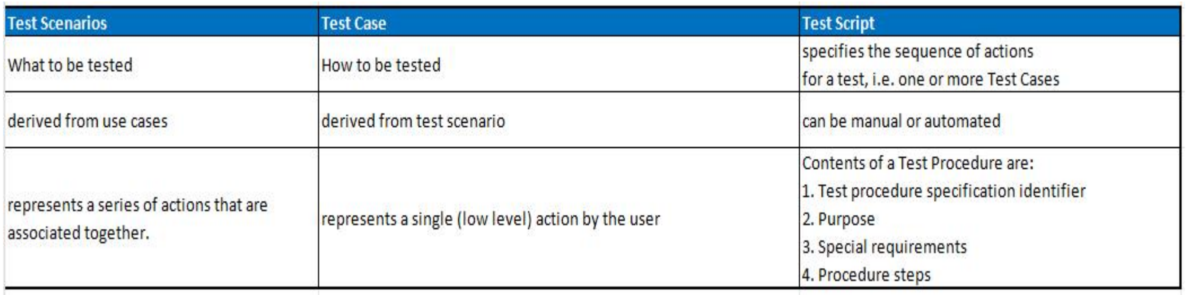
➢ https://docs.google.com/spreadsheets/d/1ctzaOhUIIG79lRlV88LVVDBCI-BKcSHC/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true
➢ https://docs.google.com/spreadsheets/d/1aQVhYKtpjzomxeeEl2a8IRfPmytxD6SK/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software- development Life Cycle)?**



Parameter	SDLC	STLC
Origin	Development Life Cycle	Testing Life Cycle
Objective	The main object of SDLC life cycle is to complete successful development of the software including testing and other phases.	The only objective of the STLC phase is testing.
Requirement Gathering	In SDLC the business analyst gathers the requirements and create Development Plan	In STLC, the QA team analyze requirement documents like functional and non-functional documents and create System Test Plan
High & Low-Level Design	In SDLC, the development team creates the high and low-level design plans	In STLC, the test analyst creates the Integration Test Plan
Coding	The real code is developed,and actual work takes place as per the design documents.	The testing team prepares the test environment and executes them
Maintenance	SDLC phase also includes post-deployment supports and updates.	Testers, execute regression suits, usually automation scripts to check maintenance code deployed.

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



Test Scenarios	Test Case	Test Script
What to be tested	How to be tested	specifies the sequence of actions for a test, i.e. one or more Test Cases
derived from use cases	derived from test scenario	can be manual or automated
represents a series of actions that are associated together.	represents a single (low level) action by the user	Contents of a Test Procedure are: 1. Test procedure specification identifier 2. Purpose 3. Special requirements 4. Procedure steps

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

- A test plan is a document that consists of all future testing-related activities. It is prepared at the project level and in general, it defines work products to be tested, how they will be tested, and test type distribution among the testers
- Test plan covers information like Objective, Scope, Testing Methodology, Approach, Risk, Mitigation plans, Defect Tracking, Test Environments, Entry and exit criteria, Test Automation.

- **What is priority?**

- 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.

- **What is severity?**

- Severity is absolute and Customer-Focused. It is the extent to which the defect can affect the software. In other Word's it defines the impact that a given defect has on the system.

- **Bug categories are…**

- Bug is an error detected in the development environment during testing stage.
- Some of bug categories are,

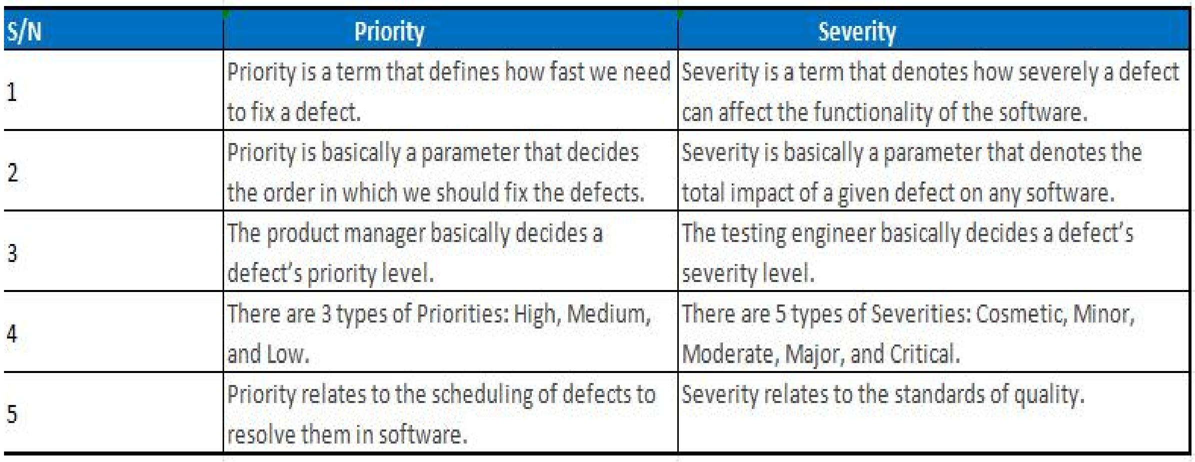
    - Functional Bugs.

- Logical Bugs.
- Workflow Bugs.
- Unit Level Bugs.
- System-Level Integration Bugs.
- Out of Bound Bugs.
- Security Bugs.

- **Advantage of Bugzila.**

- Effective Bug Tracking.
- Bugzilla provides a centralized platform for tracking and managing software defects. ...
- Customization and Flexibility. Bugzilla is highly customizable, allowing organizations to tailor it to their specific needs. ...
- Integrated Collaboration.

- **Difference between priority and severity**



S/N	Priority	Severity
1	Priority is a term that defines how fast we need to fix a defect.	Severity is a term that denotes how severely a defect can affect the functionality of the software.
2	Priority is basically a parameter that decides the order in which we should fix the defects.	Severity is basically a parameter that denotes the total impact of a given defect on any software.
3	The product manager basically decides a defect's priority level.	The testing engineer basically decides a defect's severity level.
4	There are 3 types of Priorities: High, Medium, and Low.	There are 5 types of Severities: Cosmetic, Minor, Moderate, Major, and Critical.
5	Priority relates to the scheduling of defects to resolve them in software.	Severity relates to the standards of quality.

- **What are the different Methodologies in Agile Development Model?**

- In Agile development, there are several methodologies for software testing, each with its own approach to testing within the iterative and incremental Agile framework. Here are some of the common methodologies:
- **Scrum Testing:** Scrum is one of the most popular Agile methodologies. In Scrum, testing is integrated into each sprint cycle. Testers work closely with developers to ensure that features are thoroughly tested within the short sprint duration.
- **Extreme Programming (XP):** XP emphasizes continuous testing throughout the development process. Test-driven development (TDD) is a core practice in XP, where tests are written before the actual code. XP also encourages pair programming, where one programmer writes the code while the other writes tests, ensuring high test coverage.
- **Kanban Testing:** Kanban is focused on visualizing the workflow and limiting work in progress. In Kanban, testing is performed continuously as work items move through different stages of the development process. Testers collaborate with other team members to ensure that each work item meets the required quality standards before it's considered done.

- **Feature-Driven Development (FDD):** FDD is an iterative and incremental approach to software development. In FDD, testing is integrated into each feature development cycle. Testers work closely with domain experts and developers to define acceptance criteria and ensure that each feature meets the specified requirements.
- **Crystal Methods:** Crystal methodologies prioritize teamwork and communication. Testing in Crystal is performed collaboratively by the entire team, including testers, developers, and other stakeholders. Testing activities are adapted based on the project's size, criticality, and other factors.
- **Lean Software Development:** Lean principles emphasize delivering value to customers while minimizing waste. In Lean development, testing focuses on validating assumptions and delivering working software quickly. Testers collaborate with the team to identify and eliminate waste in the testing process.
- **Dynamic Systems Development Method (DSDM):** DSDM is an Agile framework that emphasizes the importance of user involvement and frequent delivery of working software. Testing in DSDM is iterative and collaborative, with a focus on ensuring that the delivered software meets user needs and quality standards.

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

- Authorization and authentication are two distinct but related concepts in web testing:
- **Authentication:** Authentication is the process of verifying the identity of a user or system attempting to access a web application or resource. It ensures that the user is who they claim to be. Authentication mechanisms commonly used in web applications include username/password authentication, token-based authentication (such as JSON Web Tokens or JWTs), OAuth, and federated authentication (e.g., using social media accounts like Google or Facebook to log in).
- **Authorization:** Authorization, on the other hand, is the process of determining what actions a user or system is allowed to perform within the web application or on specific resources. Once a user has been authenticated, authorization determines the level of access or permissions they have. This can involve roles-based access control (RBAC), where users are assigned specific roles (e.g., admin, user, guest) with corresponding permissions, or attribute-based access control (ABAC), where access decisions are based on various attributes of the user, resource, and environment.
- Common problems faced in web testing include:
- **Broken Links and Navigation:** Testers need to ensure that all links within the web application are functional and lead to the correct pages. Broken links or incorrect navigation can result in a poor user experience and impact SEO.
- **Cross-browser Compatibility:** Web applications need to be tested across different web browsers (e.g., Chrome, Firefox, Safari, Edge) and versions to ensure consistent behavior and appearance across platforms. Variations in browser rendering engines can lead to layout issues and functionality discrepancies.
- **Security Vulnerabilities:** Web applications are susceptible to various security threats, such as SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), and insecure direct object references (IDOR). Testers must conduct security testing to identify and mitigate these vulnerabilities, ensuring the confidentiality, integrity, and availability of the application and its data.
- **Performance and Scalability:** Testers need to assess the performance of the web application under various conditions, including different levels of user traffic and concurrent users. Performance testing helps identify bottlenecks, such as slow page load times or inefficient database queries, and ensures that the application can handle increased load without degradation.

- **Usability and Accessibility:** Testing should address the usability and accessibility of the web application to ensure that it is intuitive and easy to use for all users, including those with disabilities. Usability testing evaluates the user interface and interaction design, while accessibility testing ensures compliance with accessibility standards such as WCAG (Web Content Accessibility Guidelines).

- **To create HLR & Test Case of web based (Whatsapp web , Instagram)**

➢ https://docs.google.com/spreadsheets/d/1aQVhYKtpjzomxeeEl2a8IRfPmytxD6SK/edit?usp=drive_link&ouid=113634907296333154363&rtpof=true&sd=true

- **To create HLR and TestCase on this Link. https://artoftesting.com/**

➢ https://docs.google.com/spreadsheets/d/1rlYZpEIbLC1jkHuU7L0Pc5gkGSvBgzvS/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of Pen**

➢ https://docs.google.com/spreadsheets/d/1BGCjK0Kn36GQvlDzId1Fp_WtRWNwvNO_/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of Door**

➢ https://docs.google.com/spreadsheets/d/1Ld9CaS6oGZuUDzIpblQ3sI08ddoH4XTu/edit?usp=drive_link&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of ATM**

➢ https://docs.google.com/spreadsheets/d/1KRiglBeyrWKXyRQ90aITLq68k8wRvCCu/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **When to used Usablity Testing?**

- In software testing, usability testing is utilized to evaluate how easy and intuitive it is for users to interact with the software. Here are several key scenarios when usability testing should be employed in the software testing process:

➢ Early Development Stages
➢ During Development
➢ Pre-Release
➢ Post-Release
➢ Special Circumstances

- **What is the procedure for GUI Testing?**

- GUI testing, or Graphical User Interface testing, focuses on verifying that the graphical interface of a software application functions correctly and meets user expectations. Here's a general procedure for conducting GUI testing:

➢ Understanding Requirements
➢ Identifying Test Scenarios
➢ Preparing Test Environment

- Test Case Design
- GUI Element Verification
- Functional Testing
- Navigation Testing
- Layout and Design Consistency
- Input Validation
- Accessibility Testing
- Cross-Browser and Cross-Platform Testing
- Usability Testing
- Localization and Internationalization Testing
- Documentation and Reporting
- Regression Testing

- **Write a scenario of Microwave Owen**

  - https://docs.google.com/spreadsheets/d/1hfxDmUegqUBNzyF7WjP-W0-62iJQaCMr/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a scenario of Coffee vending Machine**

  - https://docs.google.com/spreadsheets/d/1tF0ZSgTUVuU-R40ZNxDXDM6Cei4nV32g/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a scenario of chair**

  - https://docs.google.com/spreadsheets/d/14i4UtF-WGvDRxn0PjceHXxHQdO1itmln/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of Wrist Watch**

  - https://docs.google.com/spreadsheets/d/1bhL1CnpAYMRweSX0YUgU1QotCO1KTTkj/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of Lift(Elevator)**

  - https://docs.google.com/spreadsheets/d/1ESl-pUDlJ0F4W4MotHAGMlH89N1pfM4P/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of whatsapp Group (generate group)**

  - https://docs.google.com/spreadsheets/d/1uI6w_d9aq2hgohPXv9IIoHIrt4H-ZwRH/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true

- **Write a Scenario of Whatsapp payment**

  - https://docs.google.com/spreadsheets/d/1jCAJjleCYitecidR5cA_1FWc8TW1Ehh-/edit?usp=sharing&ouid=113634907296333154363&rtpof=true&sd=true