MODULE -2 (MANUAL TESTING)

**What is Exploratory Testing?**

* Exploratory testing involves concurrent test case design and test execution of an application or system under test. The tester will create or write down a test idea to give direction, and explore the system while testing to further create critical, practical, and useful tests for the successful testing of an application.
* Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking. Automation has its limits.
* Is not random testing but it is Adhoc testing with purpose of find bugs.
* Is not a technique but it is an approach. What actions you perform next is governed by what you are doing currently.

**What is Traceability Matrix?**

* When test conditions link back as the test base it is known as Traceability Matrix. Traceability Testing is basically a graph where on the horizontal base the test cases and test bases are provided and on Vertical base the development documents are provided.
* You should be able to track back from every component to the original requirements that caused its presence.

**What is Boundary Value Testing?**

* Boundary Value Analysis is based on testing the boundary values of Valid and Invalid partitions. The result at the edge of each equivalence partition is more likely to be incorrect than the result within the partition. So basically, boundaries are and area where testing is likely to yield defects.
* Every partition has its maximum and minimum values are the boundary values of a partition.
* A boundary value for a valid partition is a valid boundary value.
* A boundary value for an invalid partition is an invalid boundary value.

**What is Equivalence Partitioning Testing?**

* Equivalence Partitioning Testing is a test design testing which is used in white box and black box testing.
* It involves to dividing the entire possible range into partitions.
* It is to test each partition once.
* If you pick up any value from a partition if it is an invalid partition, any value if you pickup from that partition the system should reject that value and if you pickup from valid partition, it should accept that value.
* For an example, There is a date field and you have to enter value between 1 and 31. So here there are 3 partitions :

|  |  |  |
| --- | --- | --- |
| **INVALID PARTITION** | **VALID PARTION** | **INVALID PARTITION** |
| **0** | **1** | **32** |
| **-1** | **2** | **33** |
| **-2** | **3** | **34** |
| **-3** | **4** | **35** |
| **….** | **…..** | **……** |
|  | **31** |  |

Now, we should pickup 1 value from each partition. So when we pick 1 value from the invalid partition i.e. -1, So in this, system should reject the value because -1 is not a acceptable date in within the month.

Now , when we pick up 1 value from the valid partition i.e. 3/10, In this system should accept the value because 1 comes under dates in the month.

Now again, when we pick up 1 value from invalid partition i.e. 35 / 40, In this, system should reject the value because 35/40 is not a acceptable date within the month.

* Equivalence partition allows you to find more defects because you are testing out each and every partition and it also saves time because you are going to pickup 1 value from each partition whether the system accepts it or rejects it depending on whether it is a valid or invalid partition.

**What is Integration Testing ?**

* When Individual units are combined and tested as a group is known as Integration Testing. Testing is performed to expose defects in the interfaces and in the integration between integrated components or system.
* In integration testing, Tester and Developer both can conduct testing.
* Types / Level of Integration Testing.

1. Component Integration Testing
2. System Integration Testing

* *Component Integration Testing :* It is done after component integration testing and before system testing. This means if 2 or more components of the system merge and tested as a group to expose the defects in the interface and integration component of the system.
* *System Integration Testing :* It is done after system testing and before the acceptance testing. Here, 2 or more different systems are combined and tested to expose the defects between the interaction of 2 different system.

**What determines the level of Risk ?**

* Risk – ‘A factor that could result in future negative consequences; usually expressed as impact and likelihood’
* When testing does find defects, the Quality of the software system increases when those defects are fixed
* Risks are of two types :

1. Project Risk
2. Product Risk

**PROJECT RISK :**

* Example of Project risk is Senior Team Member leaving the project abruptly. Every risk is assigned a likelihood i.e. chance of it occurring, typically on a scale of 1 to 10. Also the impact of that risk is identified on a scale of 1- 10 . But just identifying the risk is not enough. You need to identify mitigation. In this case mitigation could be Knowledge Transfer to other team members & having a buffer tester in place

**PRODUCT RISK :**

* Example of product risks would be Flight Reservation system not installing in test environment. Mitigation in this case would be conducting a smoke or sanity testing. Accordingly you will make changes in your scope items to include sanity testing

**What is Alpha Testing?**

* Alpha testing is conducted in the organization and tested by a representative group of end-users at the developer's side and sometimes by an independent team of testers. Alpha testing is simulated or real operational testing at an in-house site. It comes after the unit testing, integration testing, etc. Alpha testing used after all the testing are executed.

**What is Beta Testing?**

* Beta testing is a type of User Acceptance Testing among the most crucial testing, which performed before the release of the software. Beta Testing is a type of Field Test. This testing performs at the end of the **software** testing life cycle. This type of testing can be considered as external user acceptance testing. It is a type of salient testing. Real users perform this testing. This testing executed after the alpha testing. In this the new version, beta testing is released to a limited audience to check the accessibility, usability, and functionality, and more.

**What is component testing?**

* Component Testing – The testing of individual software components.
* Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.
* Unit testing is the first level of testing and is performed prior to Integration Testing.
* Unit testing is performed by using the White Box Testing method.
* It is done after Component Integration testing and before system testing.
* 2 or more component of the system merge and tested as a group to expose the defects in the interfaces and integration component of the system.

**What is functional system testing?**

* 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. In functional testing, 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.
* Functional testing also called as black-box testing, because it focuses on application specification rather than actual code. Tester has to test only the program rather than the system.
* TYPES OF FUNCTIONAL SYSTEM TESTING :

1. Black Box Testing.
2. White Box Testing.
3. Experience Based Testing.
4. Smoke Testing.
5. Sanity Testing.
6. End to End Testing.

**What is Non Functional Testing ?**

* Non-functional testing is a type of software testing to test non-functional parameters such as reliability, load test, performance and accountability of the software. The primary purpose of non-functional testing is to test the reading speed of the software system as per non-functional parameters. The parameters of non-functional testing are never tested before the functional testing.
* Non-functional testing is also very important as functional testing because it plays a crucial role in customer satisfaction.
* TYPES OF NON-FUNCTIONAL TESTING :

1. Usability Testing
2. Compatibility Testing
3. GUI Testing
4. Security Testing
5. Performance Testing
6. Stress Testing
7. Load 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.

**THE WORK OF GUI TESTING IS AS GIVEN BELOW :**

* Check all the GUI elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
* Check you can execute the intended functionality of the application using the GUI
* Check Error Messages are displayed correctly
* Check for Clear demarcation of different sections on screen
* Check Font used in application is readable
* Check the alignment of the text is proper
* Check the Colour of the font and warning messages is aesthetically pleasing
* Check that the images have good clarity
* Check that the images are properly aligned
* Check the positioning of GUI elements for different screen resolution.

**What is Adhoc Testing ?**

* This testing we do when the build is in the checked sequence, then we go for Adhoc testing by checking the application randomly.
* Adhoc testing is also known as **Monkey testing and Gorilla testing**.
* It is negative testing because we will test the application against the client's requirements.

*EXAMPLE OF ADHOC TESTING :*

* Suppose we are using two different browsers like Google Chrome and Mozilla Firefox and login to the Facebook application in both the browsers.
* Then, we will change the password in the Google Chrome browser, and then in another browser (Firefox,) we will perform some action like sending a message.
* It should navigate to the login page, and asking to fill the login credentials again because we change our credentials in another browser (Chrome), this process is called adhoc testing.
* When the product release to the market, we go for Adhoc testing because the customer never uses the application in sequence/systematically for that sake; we check the application by going for Adhoc testing by checking randomly.
* Checking the application randomly without following any sequence or procedure since the user doesn't know how to use the application, they may use it randomly and find some issues to cover this we do one round of Adhoc testing.

**What is Load Testing ?**

* In software testing, load testing is an integral part of performance testing under [non-functional testing](https://www.javatpoint.com/non-functional-testing).
* Load testing is testing where we check an application's performance by applying some load, which is either less than or equal to the desired load.
* Here, load means that when **N-number** of users using the application simultaneously or sending the request to the server at a time.
* Load testing will help to detect the maximum operating capacity of an application and any **blockages** or bottlenecks.
* It governs how the software application performs while being accessed by several users at the same time.
* The load testing is mainly used to test the **Client/Server's performance and applications that are web-based.**
* In other words, we can say the **load testing** is used to find whether the organization used for compering the application is necessary or not, and the performance of the application is maintained when it is at the maximum of its user load.
* Generally, load testing is used to signify how many concurrent users handle the application and the application's scale in terms of **hardware, network capacity** etc.

**What is stress Testing ?**

* **Stress Testing** is testing used to check the accessibility and robustness of software beyond usual functional limits. It mainly considers for critical software but it can also be used for all types of software applications.
* It is also known as **Endurance Testing, fatigue testing** or *Torture Testing*.
* The stress testing includes the **testing beyond standard operational size**, repeatedly to a **breaking point**, to get the outputs.
* It highlights the error handling and robustness under a heavy load instead of correct behaviour under regular conditions.
* In other words, we can say that **Stress testing** is used to verify the constancy and dependability of the system and also make sure that the system would not crash under disaster circumstances.
* To analyses how the system works under extreme conditions, we perform **stress testing** outside the normal load.

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

* **White box testing** techniques analyse the internal structures the used data structures, internal design, code structure, and the working of the software rather than just the functionality as in black box testing. It is also called glass box testing or clear box testing or structural testing. White Box Testing is also known as transparent testing or open box testing.
* White box testing is a software testing technique that involves testing the internal structure and workings of a software application. The tester has access to the source code and uses this knowledge to design test cases that can verify the correctness of the software at the code level.

**TYPES OF WHITE BOX TESTING :**

The different types of coverage:

* STATEMENT COVERAGE
* DECISION COVERAGE
* CONDITION COVERAGE
* **STATEMENT COVERAGE :**

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

The statement coverage can be calculated as shown below:

*Statement Coverage = Number of statement exercised \* 100%*

*Total number of statements*

* **DECISION COVERAGE :**

Decision coverage also known as branch coverage or all-edges coverage. It covers both the true and false conditions unlikely the statement coverage.

The decision coverage can be calculated as shown below:

*Decision Coverage = Number of decision outcomes exercised \* 100*

*Total number of decision outcomes*

**What is Black Box testing ? What are the different black box testing techniques ?**

* Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding. The primary source of black box testing is a specification of requirements that is stated by the customer.
* In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not. If the function produces correct output, then it is passed in testing, otherwise failed. The test team reports the result to the development team and then tests the next function. After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.
* **BLACK BOX TESTING TECHNIQUES :**

1. Equivalence Partitioning
2. Boundary Value Analysis
3. Decision Table
4. State Transition Testing
5. Use-Case Testing
6. Other Black Box Testing
7. Syntax or Pattern Testing

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

There are 5 types of defect categories :

1. Data Quality / Database Defects
2. Critical Functionality Defects
3. Functionality Defects
4. Security Defects
5. User Interface Defects

**DATA QUALITY/ DATABASE DEFECTS :**

This defects means it is having a defect of the database storage. It is not getting saved properly. The values are not inserted / deleted into database properly.

**CRITICAL FUNCTIONALITY DEFECTS :**

This defects means that any main function of the application is not working properly. If the main function is not working it comes under critical functionality.

**FUNCTIONALITY DEFECTS :**

This defects means if any of the functionality of the application is not working, it is known as functionality defects. Buttons like Save, Delete, Cancel not performing their intended functions. A missing functionality (or) a feature not functioning the way it is intended to continuous execution of loops.

**SECURITY DEFECTS :**

This defects means the improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix. This includes :

Authentication : Accepting an invalid username / password.

Authorization : Accessibility to pages though permission not given.

**USER INTERFACE DEFECTS :**

As the name suggests, the bug deal with problems related to UI are usually considered less severe.

EXAMPLE :

* Improper error / warning / UI messages
* Spelling Mistakes
* Alignment problems

**Mention what big bang testing is ?**

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

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

**PURPOSE OF EXIT CRITERIA :**

* Successful Testing of Integrated Application.
* Executed Test Cases are documented.
* All High prioritized bugs fixed and closed.
* Technical documents to be submitted followed by release Notes.

**When should “Regression Testing” be performed ?**

* Testing of a previously tested programme following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environmental is changed.
* You also need to ensure that the modification have not caused unintended side effects elsewhere and that the modified system still meets its requirements.
* When there is a change in requirements and code is modified according to the requirement we need to do REGRESSION TESTING.
* When New Features is added to the software.
* Defect Fixing and Performance Issue Fix we need to perform Regression Testing.

**What is the 7 principles? Explain in detail ?**

**7 Key Principle :**

1. Testing shows presence of defects.
2. Exhaustive Testing is Impossible.
3. Early Testing.
4. Defect Clustering.
5. The Pesticide Paradox.
6. Testing is context dependent.
7. Absence of errors fallacy.

**TESTING SHOWS PRESENCE OF DEFECTS :**

* Testing can show that defects are present, but cannot prove that there are no defects. Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.
* As we find more defects, the probability of undiscovered defects remaining in a system reduces. However Testing cannot prove that there are no defects present.

**EXHAUSTIVE TESTING IS IMPOSSIBLE :**

* Testing everything including all combinations of inputs and preconditions is not possible.
* Exhaustive testing of complex software applications:

1. Requires enormous resources
2. It is too expensive
3. It takes too long

**EARLY TESTING :**

* Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives.
* Testing activities should start as early as possible in the development life cycle. These activities should be focused on defined objectives – outlined in the Test Strategy.

**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’.

**PESTISIDE PARADISE :**

* If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects.
* To overcome this “pesticide paradox”, the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects.
* We must learn, create and use new tests based on new techniques to catch new bugs.

**TESTING IS CONTEXT DEPENDENT :**

* Testing is basically context dependent. Testing is done differently in different contexts.
* Different kinds of sites are tested differently.
* For an example, if there is a banking application we will see that application from security aspects. In the same way when there is a gaming application in that we will see about the graphics used in that application.
* This means according to the concern application the testing is conducted.

**ABSENCE OF ERROR FALLACY :**

* If the system built is unusable and does not fulfil 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.**

|  |  |  |
| --- | --- | --- |
| **QUALITY ASSURANCE** | **QUALITY CONTROL** | **TESTER** |
| 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. |
| Focuses on processes and procedures rather than conducting actual testing | Focuses on actual testing by executing Software with intended to identify bug/defect through implementation of procedures and process | Focuses on actual testing |
| Process oriented activities | Product oriented activities | Product oriented activities |
| Preventive Activities | It is a corrective process. | It is a preventive process. |
| 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 ?**

|  |  |
| --- | --- |
| **SMOKE TESTING** | **SANITY TESTING** |
| Smoke testing is a shallow and wide approach of testing. | Sanity testing is a narrow and deep approach of testing. |
| Smoke testing is done to ensure that the build is good enough to be considered for carrying on full-fledged testing. | Sanity testing is done to ensure that after a new fix a particular component or feature of the application is working fine. |
| We usually have documented smoke tests i.e. there are some tests identified as smoke tests. These are tests are usually(preferably) automated. | Sanity tests are generally not automated or even documented as test cases. |
| Smoke checks the overall components of the system. | Sanity checks the specific functionality of the system. |
| It can be considered as a general health checkup of application under test | It can be considered as a specific heal checkup of the application under test |

**Difference between verification and validation.**

|  |  |
| --- | --- |
| **VERIFICATION** | **VALIDATION** |
| 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. |
| 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 users need, and that the specifications were correct in the first place. In other words, to demonstrate that the product fulfils its intended use when placed in its intended environment. |
| Plans, Requirement Specs, Design Specs, Code, Test Cases | The actual product / software |

**What is Error, Defect, Bug and Failure ?**

**Error :** An error is a mistake made by human that leads to discrepancy between the actual and the expected result.

**Defect :** A defect is a problem in the functioning of a software system during testing. [ISTQB](https://www.astqb.org/glossary/) defines a defect as “A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g., an incorrect statement or data definition.”

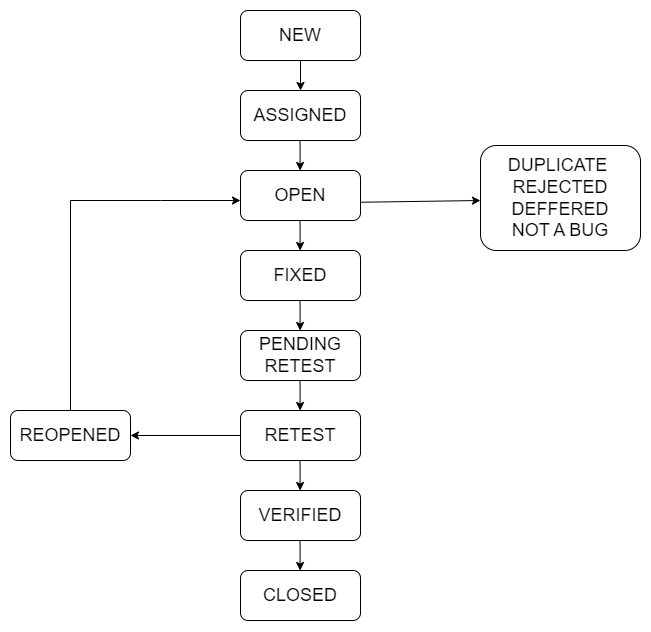
**Bug :** A bug is a flaw in a software system that causes the system to behave in an unintended manner.

**Failure :** A failure is the inability of a software system to perform its operations within the specified performance benchmark. As per [ISTQB](https://www.astqb.org/glossary/), “a defect, if encountered during execution, may cause a failure of the component or system”.

**Difference between Priority and Severity**

**What is Bug Life Cycle ?**

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 ‘Defect Life Cycle’. The process by which the defect moves through the life cycle is given below.



**Explain the difference between Functional Testing and Non Functional Testing.**

|  |  |
| --- | --- |
| **FUNCTIONAL TESTING** | **NON-FUNCTIONAL TESTING** |
| 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. |
| Functional Testing is executed first | Non-functional testing should be performed after functional testing |
| Manual testing or automation tools can be used for functional testing | Using tools will be effective for this testing |
| Business Requirements are the inputs to functional testing | Performance parameters like speed, scalability are inputs to non-functional testing |
| Functional testing describes what the product does | Non-Functional testing describes how good the product works |
| Easy to do manual testing | Tough to do manual testing |
| Types of Functional testing are   1. Unit Testing 2. Smoke Testing 3. Sanity Testing 4. Integration Testing 5. White Box Testing 6. Black Box Testing 7. User Acceptance  8. Regression Testing | Types of Non-Functional testing are  1. Performance Testing 2. Load Testing 3. Volume Testing 4. Stress Testing 5. Security Testing 6. Installation Testing 7. Penetration Testing 8. Compatibility Testing 9. Migration Testing |

**To Create HLR & Test Case of**

1. **(Instagram, Facebook) only first page**
2. **Facebook Login Page**

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

* SDLC stands for SOFTWARE DEVEOPMENT LIFE CYCLE. It describes the various phases involved in the software development process. With the of SDLC, we can create software application in a well defined and systematic way.
* STLC stands for SOFTWARE TESTING LIFE CYCLE. It refers to all these activities performed during the testing of a software product. Basically, it provides a sequence of activities performed to ensure the quality of the software application.

|  |  |
| --- | --- |
| **SDLC PHASES** | **STLC PHASES** |
| Requirement Gathering | Requirement Analysis |
| Analysis | Test Planning |
| Design | Test Case Development |
| Implementation | Test Environment Setup |
| Testing | Test Execution |
| Maintenance | Test Cycle Closure |

The different phases of the Software Development Life Cycle are-

* Requirement Gathering and Analysis – In this phase, all the requirements are gathered and analysed for their feasibility.
* Designing – In this phase, the requirement specifications are converted into design specifications.
* Coding/Implementation – Actual coding is done here.
* Testing – This phase involves testing of the software product.
* Deployment – The software is deployed to production for the end-user.
* Maintenance – Due to change in the environment and for continuous improvement maintenance is required.

The different phases of Software Testing Life Cycle are-

* Requirement Analysis – In this phase, the high-level analysis of the requirements is done.
* Test Planning – In this phase, a test strategy and approach are defined.
* Test Case Development – The test cases are created in this phase.
* Test Environment Setup – Here, the test environment is created in which the test execution will be performed.
* Test Execution – Test cases are executed and defects are logged for the failed tests.
* Test Closure – A test closure document is prepared which contains all the testing activities performed and the bugs found. This phase marks the formal closure of the testing phase.

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

**TEST SCENARIO :** A test scenario is any functionality that can be tested. It is also called TEST CONDITION or Test Possibility. Test Scenario means to look after both the points, Positive as well as negative aspects of each functionality. Test Scenario is nothing but a test procedure.

**TEST CASES :** Test cases involve the set of steps, conditions and input which can be used while performing the testing tasks. Test Case is “How to be tested”. Test case consist of set of input values, execution preconditions, expected results and executed post condition developed to cover certain test Conditions.

**TEST SCRIPTS :** A set of sequential instruction that detail how to execute a core business function. A test script in software testing is a set of instructions that will be performed on the system under test to test that the system functions as expected. There are various means for executing test scripts, Manual and Automation.

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

**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. 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.

**What is severity ?**

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. 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.

**Bug categories are …**

There are 4 categories of Bug :

1. High Priority High Severity
2. Hight Priority Low Severity
3. Low Severity High Priority
4. Low Severity Low Priority

**Advantage of Bugzila.**

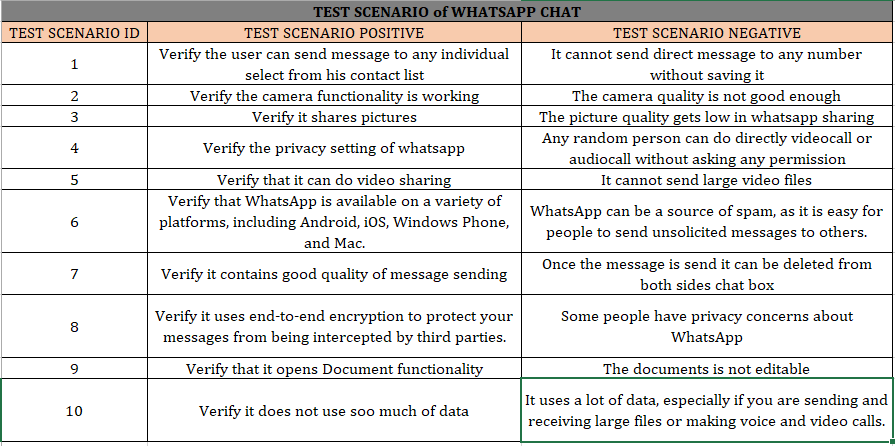
**What are the different Methodologies in Agile Development Model ?**

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

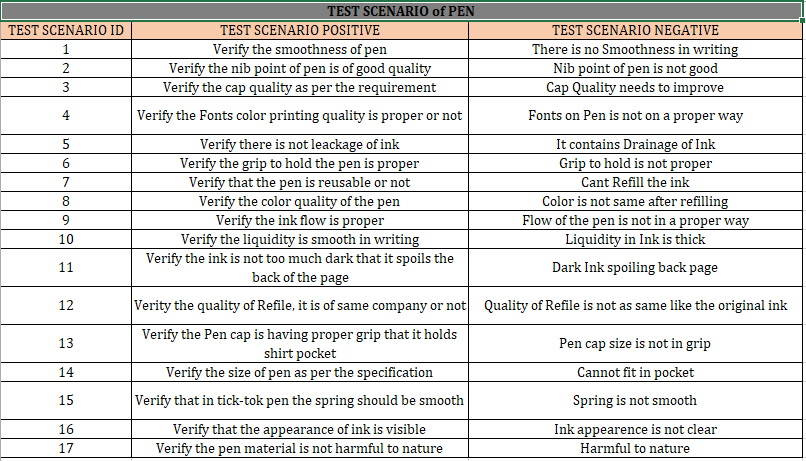
**To create HLR & Test Case of Web based :**

1. **WHATSAPP WEB.**
2. **INSTAGRAM WEB.**
3. **ART OF TESTING**

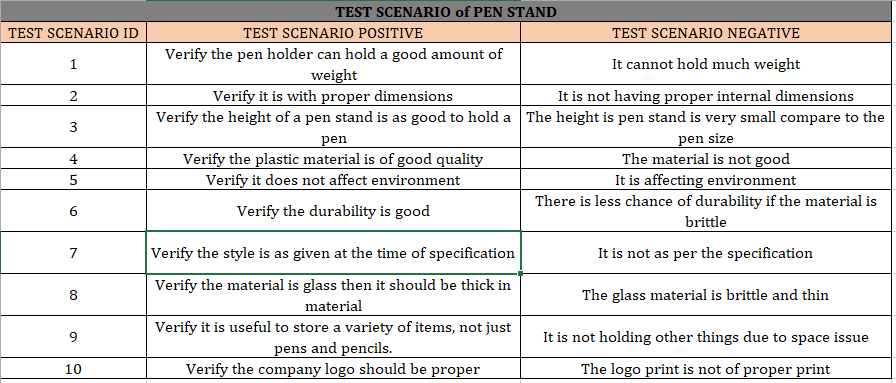
**Write a scenario of only Whatsapp Chat messages**

**­­­­­­**

**Write a scenario of Pen**

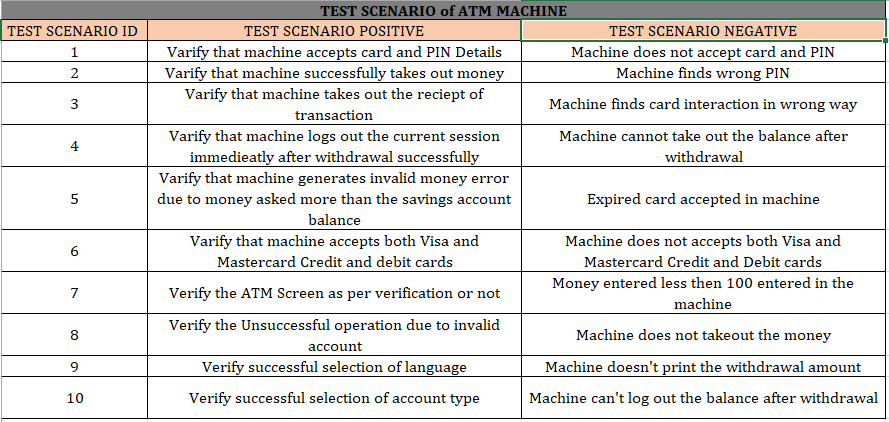
****

**Write a scenario of Pen Stand**

****

**Write a scenario of Door**

**Write a scenario of ATM**

****

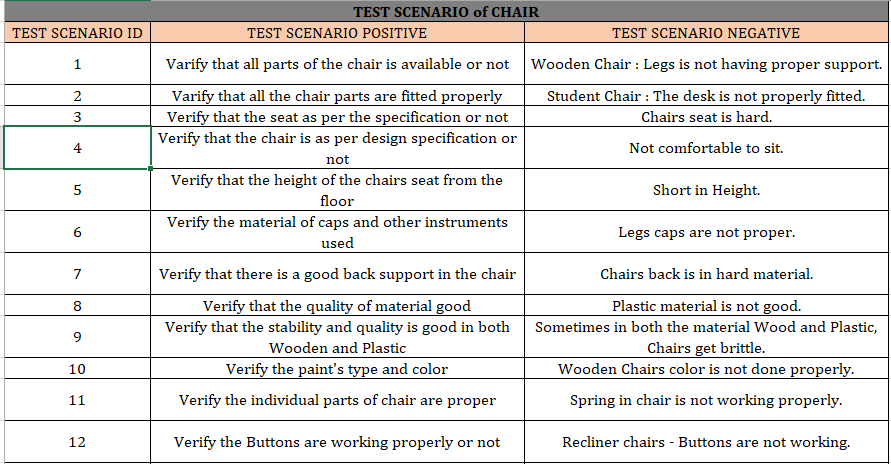
**When to used Usability Testing ?**

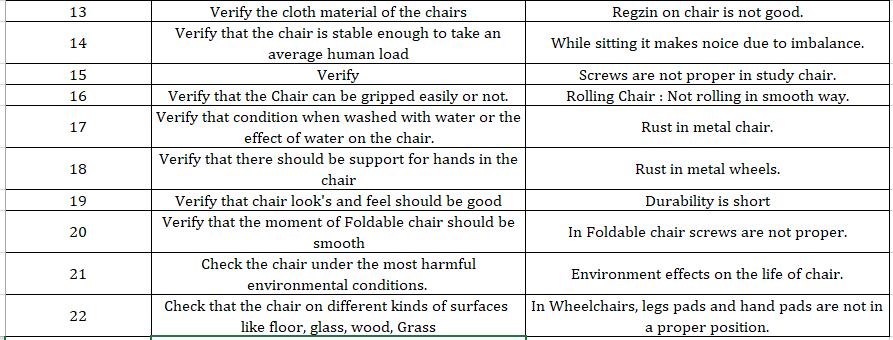
**What is the procedure for GUI Testing ?**

**Write a scenario of Microwave Owen**

**Write a scenario of Coffee Vending Machine**

**Write a scenario of chair**

****

****

**Write a scenario of Facebook Chat on Mobile**

**Write a scenario of Gmail**

**Write a scenario of Online shopping to buy product Flip cart**

**Write a scenario of Wrist Watch**

**Write a scenario of Lift (Elevator)**

**Write a scenario of Whatsapp group (generate group)**

**Write a scenario of Instagram (video call with chat)**

**Write a scenario of Whatsapp Payment**