**Software Testing Assignment**

**Module-2(manule testing)**

1. **What is Exploratory Testing ?**

**ANS :-** Exploratory testing is an approach to software testing that is often described as simultaneous learning, test design , and execution.

1. **What is traceability matrix ?**

**ANS :-** A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship.

It is used to track the requirements and to check the current project requirements are met.

1. **What is Boundary value testing ?**

**ANS :-** Software testing technique in which tests are designed to include representatives of boundary values. It is performed by the QA testing teams.

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

**ANS :-** In this method the input domain data is divided into different equivalence data classes. This method is typically used to reduce the total number of test cases to a finite set of testable test cases, still covering maximum requirements. In short it is the process of taking all possible test cases and placing them into classes. One test value is picked from each class while testing.

* One input data class with all valid inputs. Pick a single value from range 1 to 1000 as a valid test case. If you select other values between 1 and 1000 then result is going to be same. So one test case for valid input data should be sufficient.
* Input data class with all values below lower limit. I.e. any value below 1, as a invalid input data test case.

**5.What is Inetgration testing ?**

**ANS :-**  System Integration Testing is testing between the ‘System’ and ‘Acceptance’ phases.

• The System has already proven to be functionally correct, what remains to be tested is how the system reacts to other systems and/or organizations.

• The objective of System Integration Testing is to provide confidence that the system or application is able to interoperate successfully with other specified software systems and does not have an adverse effect on other systems that may also be present in the live environment, or vice versa

• It is possible that the testing tasks performed during System Integration Testing may be combined with System Testing, particularly if the system or application has little or no requirement to interoperate with other systems

• In terms of the V Model, Systems Integration Testing corresponds to the Functional and Technical Specification phases of the software development lifecycle

• Having completed Component integration testing and Systems testing, one must execute the plan for system-to-system integration

• Infrastructure may need to be transformed in order to feed to an external system

• Black Box testing techniques used.

**6.what determines the level of risk ?**

**ANS :-**

• A properly designed test that passes, reduces the overall level of Risk in a system

• 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

• The Quality of systems can be improved through Lessons learned from previous projects

• Analysis of root causes of defects found in other projects can lead to Process Improvement

• Process Improvement can prevent those defects reoccurring

• Which in turn, can improve the Quality of future systems

• Testing should be integrated as one of the Quality assurance activities

**7.What is alpha testing ?**

**ANS :-** Type of testing a software product or system conducted at the developer's site. Usually it is performed by the end user.

• It is always performed by the developers at the software development site.

• Sometimes it is also performed by Independent Testing Team.

• Alpha Testing is not open to the market and public

• It is 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 organizations location with the involvement of developers.

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

• During this phase, the following will be tested in the application:

• Spelling Mistakes

• Broken Links

• Cloudy Directions

• Alpha Testing is always performed at the time of Acceptance Testing when developers test the product and project to check whether it meets the user requirements or not.

• It is always performed at the developer’s premises in the absence of the users.

• It is considered as the User Acceptance Testing (UAT) which is done at developer’s area.

• Unit testing, integration testing and system testing when combined are known as alpha testing.

**8. What is beta testing ?**

**ANS :-** • It is always performed by the customers 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 (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.

• Beta Testing is always performed at the time when software product and project are marketed.

• It is always performed at the user’s premises in the absence of the development team.

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

• Beta testing can be considered “pre-release” testing.

• Pilot Testing is testing to product on real world as well as collect data on the use of product in the classroom.

**9. What is component testing ?**

|  |
| --- |
| **Ans.** Component testing, also known as program or module testing, is done after unit testing. In this type of testing those test objects can be tested independently as a component without integrating with other components e.g. modules, classes, objects, and programs. This testing is done by the development team.  **10. What is functional system testing?**  **Ans.** Functional Testing is a type of Software Testing in which the system is tested against the functional requirements and specifications. Functional testing ensures that the requirements or specifications are properly satisfied by the application.  **11. What is Non-Functional Testing?**  **Ans.** Non-Functional Testing is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc) of a software application.  **12. What is GUI Testing?**  **Ans :-** GUI Testing is a software testing type that checks the Graphical User Interface of the Software.  The purpose of Graphical User Interface (GUI) Testing is to ensure the functionalities of software application work as per specifications by checking screens and controls  like menus, buttons, icons, etc.  **13. What is Adhoc testing?**  **Ans** :- Ad hoc Testing is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage.  Ad hoc testing is done randomly and it is usually an unplanned activity which does not follow any documentation and test design techniques to create test cases.  **14. what is load testing ?**    **ANS.**  Testing technique that puts demand on a system or device and measures its response. It is usually conducted by the performance engineers.  **15. What is stress testing ?**  **ANS.** Testing technique which evaluates a system or component at or beyond the limits of its specified requirements. It is usually conducted by the performance engineer.  **16. What is white box testing and list the types of white box testing?**  **ANS.** White box testing is software testing technique in which internal structure, design and coding of software are tested to verify flow of input – output and to improve  Design, usability and security.  in white box testing, code is visible to testers so it is aiso called clear box testing, open box testing, transparent box testing, Code-based testing and Glass box testing.  Type :-  1. Branch Condition Testing  2. Branch Condition Combination Testing  3. Modified Condition Decision Testing  4. Dataflow Testing  5. Linear Code Sequence And Jump Testing  **17. What is black box testing? What are the different black box testing techniques?**  **ANS.** a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon. |
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|

Testing techniques

• Equivalence partitioning

• Boundary value analysis

• Decision tables

• State transition testing

• Use-case Testing

• Other Black Box Testing

• Syntax or Pattern Testing

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

**ANS.** Common Types of Defects

* Arithmetic Defects
* Logical Defects.
* Syntax Defects.
* Multithreading Defects.
* Interface Defects.
* Performance Defects

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

**ANS.** Testing technique which integrates individual program modules only when everything is ready. It is performed by the testing teams.

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

**ANS.** The purpose of exit criteria to prevent a task from being considered completed when there are still outstanding parts of the task which have not been finished . Exit criteria are used to report against and to plan when to stop testing.

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

**ANS.** Type of software testing that seeks to uncover software errors after changes to the program (e.g. bug fixes or new functionality) have been made, by retesting the program. It is performed by the testing teams.

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

ANS.

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

**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.

• We test to find Faults

• As we find more defects, the probability of undiscovered defects remaining in a system reduces.

**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.

• For example: In an application in one screen there are 15 input fields, each having 5 possible values, then to test all the valid combinations you would need 30 517 578 125 (515) tests.

• 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, and 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.Pesticide Paradox**  
 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.

**6.Testing is Context Dependent**

* Testing is basically context dependent, Testing is done differently in different contexts
* Diffrenent kinds of sites are tested differently.

**7.Absence of Errors Fallacy**

* If the system built is unusable and does not fulfill the user’s needs and expectations then finding and fixing defects does not help.  
    
   If we build a system and, in doing so, find and fix defects - It doesn’t make it a good system.  
    
  Even after defects have been resolved it may still be unusable and/or does not fulfil the users.

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

**ANS.**  While QA testing focuses on providing assurance that quality requested will be achieved, QC testing focuses on fulfilling the quality requested. QA focuses on

* preventing defect while QC focuses on identifying the defect.

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

**Ans**. Smoke testing is a subset of acceptance testing whereas Sanity testing is a subset of Regression Testing. Smoke testing is documented or scripted whereas  
  
Sanity testing isn't. Smoke testing verifies the entire system from end to end whereas Sanity Testing verifies only a particular component.

**25. Difference between verification and Validation**

**Ans.** **Verification :**

It includes checking documents, design, codes and programs.  
  
Verification is the static testing.  
  
It does not include the execution of the code.  
  
Quality assurance team does verification.  
  
It comes before validation.  
  
It consists of checking of documents/files and is performed by human.  
  
**Validation :**  
  
It includes testing and validating the actual product.  
  
Validation is the dynamic testing.  
  
It includes the execution of the code.  
  
Validation is executed on software code with the help of testing team.  
  
It comes after verification.  
  
It consists of execution of program and is performed by computer.

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

**Ans.** Load Testing :

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

**Ans.**  A mistake in coding is called Error, error found by tester is called Defect, defect accepted by development team then it is called Bug, build does not meet the requirements then it Is Failure.

**28. Difference between Priority and Severity**

Ans. Severity is a parameter to denote the impact of a particular defect on the software. Priority is a parameter to decide the order in which defects should be fixed.  
  
Severity means how severe defect is affecting the functionality. Priority means how fast defect has to be fixed.

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

**Ans.** Bug life cycle also known as defect life cycle is a process in which defect goes through different stages in its entire life. This lifecycle starts as soon as a bug is  
reported by the tester and ends when a tester ensures that the issue is fixed and won't occur again.

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

**ANS.**

* **Functional Testing**

Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements.

Functional testing is executed first

Manual testing or automation tools can be used for functional testing

Business requirements are the inputs to functional testing

Functional testing describes what the product does

Easy to do manual testing

Types of Functional testing are

• Unit Testing

• Smoke Testing

• Sanity Testing

• Integration Testing

• White box testing

• Black Box testing

• User Acceptance testing

• Regression Testing

* **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 testin

Performance parameters like speed , scalability are inputs to non-functional testing.

Nonfunctional testing describes how good the product works

Tough to do manual testing

Types of Nonfunctional testing are

• Performance Testing

• Load Testing

• Volume Testing

• Stress Testing

• Security Testing

• Installation Testing

• Penetration Testing

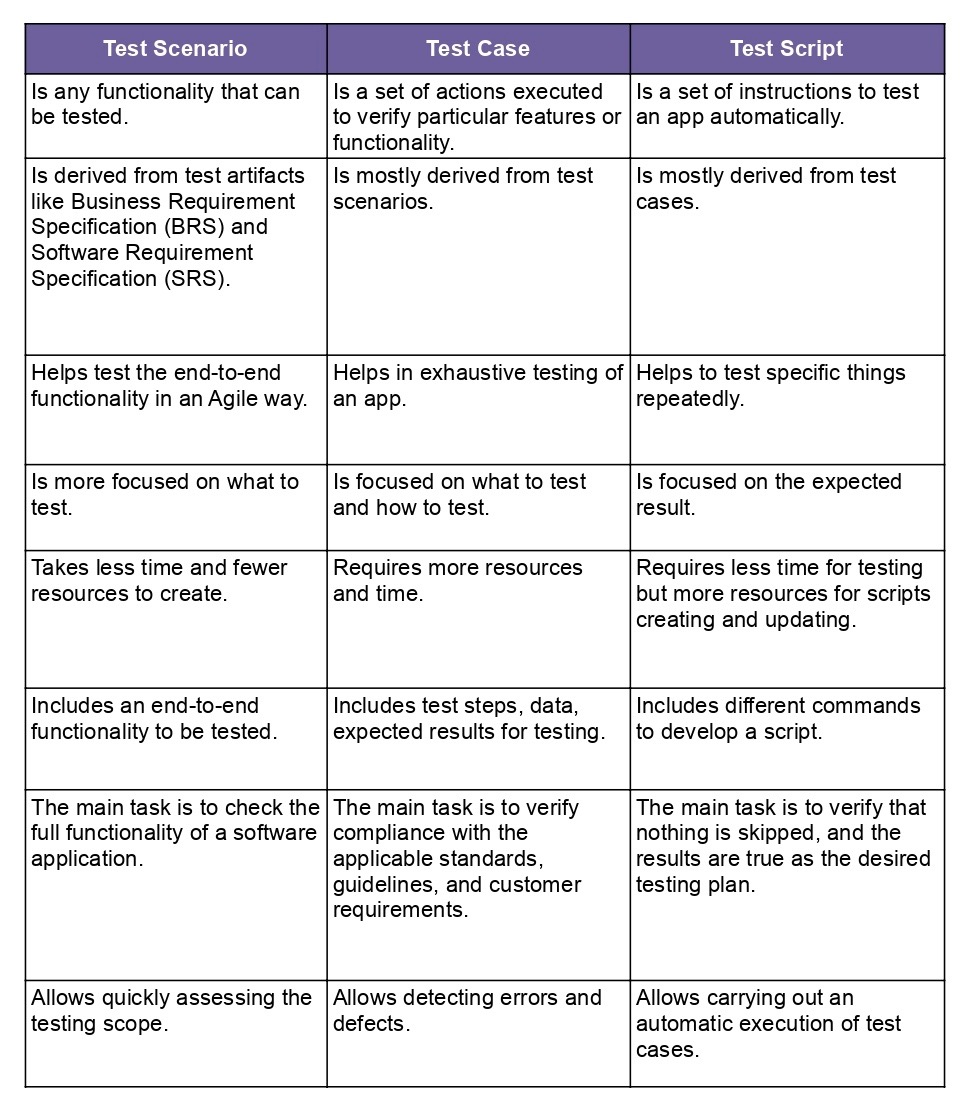
• Compatibility Testing

• Migration Testing

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

**Ans. SDLC :**  
  
SDLC is mainly related to software development.  
  
Besides development other phases like testing is also included.  
  
SDLC involves total six phases or steps.  
  
In SDLC, more number of members (developers) are required for the whole process.  
  
In SDLC, development team makes the plans and designs.  
  
Goal of SDLC is to complete successful development of software.  
381  
It helps in developing good quality software.  
  
SDLC phases are completed before the STLC phases.  
  
**STLC :**  
  
STLC is mainly related to software testing.  
  
It focuses only on testing the software.  
  
STLC involves only five phases or steps.  
  
In STLC, less number of members (testers) are needed.  
  
In STLC, testing team makes the plans and designs.  
  
Goal of STLC is to complete successful testing of software.  
  
It helps in making the software defects free.  
  
STLC phases are performed after SDLC phases.

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

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

**Ans.**  A **Test Plan** is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product. Test Plan helps us determine the effort needed to validate the quality of the application under test. The test plan serves as a blueprint to conduct software testing activities as a defined process, which is minutely monitored and controlled by the test manager.

As per ISTQB definition: “Test Plan is A document describing the scope, approach, resources, and schedule of intended test activities.”

Let’s start with following Test Plan example/scenario: In a meeting, you want to discuss the Test Plan with the team members, but they are not interested – .

## What is the Importance of Test Plan?

Making Test Plan document has multiple benefits

* Help people outside the test team such as developers, business managers, customers **understand** the details of testing.
* Test Plan **guides** our thinking. It is like a rule book, which needs to be followed.
* Important aspects like test estimation, test scope,[Test Strategy](https://www.guru99.com/how-to-create-test-strategy-document.html)are **documented** in Test Plan, so it can be reviewed by Management Team and re-used for other projects.

## How to write a Test Plan

You already know that making a **Test Plan** is the most important task of Test Management Process. Follow the seven steps below to create a test plan as per IEEE 829

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

**34. What is priority?**

**Ans.** A defect is the representation of a misunderstanding of specification or syntax error, where representation is mode of expression. Defect Priority is classification of a defect based on its importance or precedence to be fixed. Defects that force software system to become ineffectual are given higher priority over defects that cause a small functionality of software to fail.

The priority of defects is defined by business stakeholders, product owner, project managers, business analysts according to [severity](https://www.geeksforgeeks.org/defect-severity/) of defect and business needs. Priority classification is associated with software scheduling.

Defect Priority –  
Following are the different defect priority classes in software development according to end-user perspective:

1. **Immediate:**

A defect that needs to be fixed immediately as it affects system and business requirements severely. It restricts system from performing crucial functionalities and blocks execution of testing of the software, forcing software system to become ineffectual.

Any defect that impacts business requirement, needs immediate consideration and is classified as Immediate. All critical severity defects come under category of Immediate Priority but all immediate priority defects do not necessarily belong to critical severity defect. It is represented by P1.

Example –  
We have an E-commerce shopping website and company name is misspelled in home page of website, then priority of defect is high but severity is low.  
As misspelling may create a negative perception about company’s website (people may think that website is fake) leading to losses in business, but misspelling does not affect functionality of website.

1. **High:**

This defect comes after immediate as it needs less attention but still more than other remaining priority categories. These defects are resolved once critical issues are solved.  
A defect that affects business and major functionality/feature of software due to a program defect or sometimes environmental problem comes under category of High Priority. It is represented by P2.

Example –  
We have an E-commerce shopping website and it crashes whenever, we add an item to our wish list section.

1. **Medium:**

A defect with minor severity that need not be fixed right away as it does not cause any significant functionality issues or business. These defects are fixed after Immediate and High priority defects are removed. All the Minor severity defects fall into this category. It is represented by P3.  
Sometimes, even trivial errors such as wrong error message in a system can qualify for a medium priority defect.

Example –  
We have an E-commerce shopping website and we make a purchase. The purchased item is added to Order Section on website but payment portal displays wrong “Payment Unsuccessful” prompt.

1. **Low:**

A defect that does not have any major impact on functionality of software and hence does not need any immediate attention. It can be repaired in future or once higher priority defects are fixed. All the Low severity defects fall into this category. It is represented by P4.  
Sometimes, low priority defects are also used to suggest some enhancements in existing software design or to create a small feature to enhance user experience.

Example –  
We have an E-commerce shopping website and navigation bar on home screen is misaligned.

**35. What is severity?**

**Ans.** A defect is the representation of a misunderstanding of specification or syntax error, where representation is the mode of expression. Defect severity is the classification of a defect based on its level of destructive impact on the requirements specification of the Software.

Defect Severity:   
**Major –**Following are some of the commonly used defect severity classes in the software development:

1. **Critical –**  
   A defect that completely obstructs the execution of a core functionality/feature of the software is classified a critical defect. This defect affects the critical functionalities and data and makes the testing of the software difficult. It is represented by S1.

Example: We have a login interface for a mobile application. When a user taps on forgot password, the user is asked to enter the OTP which he/she receives on his registered mobile number or email. Upon entering the OTP (correct/incorrect), a loader is displayed and continues indefinitely. Since, the user is not able to login to access the functionalities of the mobile application, it is a critical defect.

1. A defect that makes a major functionality/feature to behave grossly away from what is specified in the requirements specification of the software is classified a major defect. This defect affects the major functionalities and data. It is represented by S2.

Example: We have a login interface for a mobile application and the system is allowing the access to the platform under the entered username without validating the password for its respective username. As a result, an unauthorised user can access the account leading to a major defect causing theft of personal data and criminal activity.

1. **Minor –**  
   A defect which occurs when a functionality/feature does not behave as intended or exhibits some unnatural behavior, however the functionality/feature as a whole is not much impacted is classified a minor defect. This minimally affects the functionalities and data. It is represented by S3.

Example: We have a login interface for a mobile application. The user enters his username and password, and after validation the user successfully logins to his/her account but is displayed “Login Unsuccessful Please try again later” prompt window instead. This is a minor defect as he is just shown a wrong prompt window.

1. **Trivial –**  
   Any cosmetic defects such as misplaced images, spelling mistakes or alignment issues or font casing is classified a trivial defect. This defect does not affect functionalities and data. It is represented by S4.

Example: We have a login interface for a mobile application and the login credentials have a incorrect order that is the password field is placed above the username field.

**36. Bug categories are…**

**Ans**. No matter the software type, software bugs are categorized into three types; Nature, Priority, and Severity. Classification of bugs in software testing is done on the basis of their nature and impact on the user experience.

1. **Software Bugs by Nature:**   
   Software bugs have different natures where they affect the overall functioning of the software differently. Though there are dozens of such bugs existing currently, you may not face them frequently. With that in mind, here are the most common software bugs categorized by nature that you are most likely to witness at some point in your software development career.
   * **Performance Bugs:**   
     No user wants to use software with poor performance. Software bugs that lead to degraded speed, stability, increased response time, and higher resource consumption are considered performance bugs. The most significant sign of any such bug in software is by noticing slower loading speed than usual or analysing the response time. If any such sign is found, the developer may begin diagnosing a performance bug. The [performance testing](https://www.thinksys.com/performance-testing-services/) phase is part of the development process where every such bug is detected in the software.
   * **Security Bugs:**   
     While using software, security is the biggest concern of a user. Software with poor security will not only put the user’s data at risk but will also damage the overall image of the organization which may take years to recuperate. Due to their high severity, security bugs are considered among the most sensitive bugs of all types. Though it is self-explanatory, security bugs may make the software vulnerable to potential cyber threats. Sometimes, the software organization may not notice such attacks whereas in some cases, these attacks could cause monetary loss to the users, especially small and medium-scale businesses. XSS vulnerabilities, logical errors, and encryption errors are some of the commonest security bugs found in the software. Developers put special focus on checking the code to find any underlying [security bug](https://www.thinksys.com/security-testing-services/) to minimize the risk of cyber-attacks.
   * **Unit Level Bugs:**   
     Unit level bugs are fairly common in software development and do not cause much damage to it as well. Facing basic logic bugs or calculation errors are considered unit-level bugs. The [testing team](https://www.thinksys.com/software-testing-services/) along with the agile team test a small part of the code as a whole. The reason why this testing method is preferred is to make sure that the entire code is working as it is meant to. While testing, the team may encounter unit-level bugs which can be fixed easily as the team is only working with a small code.
   * **Functional Bugs:**   
     Software is as good as the feature it provides. If any of the functionality of a software is compromised, the number of users will start to decline drastically until it becomes functional again. A functional bug is when a certain feature or the entire software is not functioning properly due to an error. The severity of such bugs depends on the feature they are hampering. For instance, an unresponsive clickable button that is not functioning is not as severe as the entire software not working. [Functional testing](https://www.thinksys.com/functional-testing-services/) is done by the testing team to identify any such software bug causing functionality errors. Once identified, the team decides its further classification and severity.
   * **Usability Bugs:**   
     Probably one of the most catastrophic bugs for software, a usability bug or defect can stop the software from working to its potential or make it entirely unusable. Examples of this bug in software testing are the inability to log in to the user account or the inefficient layout of the software for the user. The bottom line is that this type of defect or bug can make it complex for the user to use the software efficiently. The developers and engineers have to look out for the right [usability requirements](https://www.thinksys.com/qa-testing/complete-guide-to-usability-testing/) while testing the code to identify such bugs.
   * **Syntax Errors:**   
     Syntax errors are among the commonest software bug types and do not allow the application to be compiled appropriately. This bug occurs due to an incorrect or missing character from the source code due to which the compiling will be affected. A small error like a missing bracket could lead to this problem. The development or testing team will get to know about this bug during compiling and will further analyse the source code to fix the missing or wrong characters.
   * **Compatibility Errors:**   
     Whenever a software or an application is not compatible with hardware, or an operating system, it is considered as incompatible software or a compatibility error. Finding a compatibility error is not a common practice as they may not show up in the initial testing. Due to this reason, the developers should go for compatibility testing to make sure that their created software is compatible with common hardware and operating systems.
   * **Logic Bugs:**   
     Another one of the most frequently found bugs in a software code, logic errors make the software give wrong output, software crash or failure. In the majority of cases, these bugs are caused due to coding errors where it may make the software stuck in a never-ending loading loop. In that case, only an external interruption or software crashing are the two only things that can break the loading loop.
2. **Priority-Based Software Bugs:**   
   The foremost category here is priority-based software bugs. These are based on the impact these bugs leave on the business. Here, the developers will analyse the bug to determine its impact and its defect priority. Afterward, the timeline is given to each bug where it should be rectified within the stipulated time frame to minimize the bug effect on the user. Here are the four types of priority-based software bugs.
   * **Low-priority defects:**   
     Low priority defects do not cause much impact on the functioning of the application. Rather, they are more about software aesthetics. For instance, any issue with the spelling or the alignment of a button or text could be a low-priority defect. The software testing will move to the exit criteria even if the low-priority defects are not fixed, but they should be rectified before the final release of the software.
   * **Medium-priority defects:**   
     Akin to low-priority defects, medium-priority defects do not cause any significant impact on the software, but they should be fixed in any subsequent or upcoming release. Such defects may not have the same effect for every user and it may vary with the device as well as specific configuration they have.
   * **High-priority defects:**   
     Unlike the previous two, the exit criteria of high-priority defects are not met until the issue is resolved. Every bug falling in this category may make certain features of the software unusable. Even though it may not affect every user, it is mandatory to fix these bugs before any further step is taken in software development or testing.
   * **Urgent Defects:**   
     As the name suggests, all bugs that should be dealt with utmost urgency fall under this category. Urgent defects may leave a lasting impact on the brand image as well as affect the user experience drastically. The stipulated timeline for fixing these bugs is within 24-hours of reporting.
3. **Software Bugs by Severity:**   
   Depending on the technical effect that the bug will cause on the software, the bugs are categorized into four categories.
   * **Low Severity Bugs:**   
     Low severity bugs do not cause much damage to the functioning of the software as their primary target is the user interface. For instance, the font of the text on the program differs from what was used. These bugs can be fixed easily and are nothing to worry about.
   * **Medium Severity Bugs:**   
     Every bug that can affect the functionality of the software a little bit is considered a medium severity bug. All such bugs make the software function different from what it is supposed to function. Though they are not also major for the program, they should be fixed for a better user experience.
   * **High Severity Bugs:**   
     High severity bugs affect the software functionality, making it behave differently from what it was programmed for. Not only are such bugs damaging for the software, they sometimes make the entire software unusable for the user.
   * **Critical Bugs:**   
     Critical bugs are the most damaging bugs in the category that can hinder the functionality of the entire software. The reason why critical bugs are considered the most damaging is that further testing on the software becomes impossible till such bugs exist in the software.

**37. Advantage of Bugzila .**

**Ans.** There are many bug tracking systems at present. A bug tracker is an inherent part of any web site testing, desktop testing, or mobile testing. Each bug tracking instrument has its characteristics, strengths, and weaknesses.

One of the popular bug trackers is Bugzilla. Any software testing company knows that it has been on the market for more than 15 years and has a reputation of a good open-source bug racking tool.

Many testers have been working with it for years. Bugzilla became a kind of standard and other error tracking systems are often compared to it.

A lot of software testing experts find that Bugzilla is best for mobile application testing, desktop testing, or web site testing when the project is small.

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 with test management instruments;

• It integrates with an e-mailing system;

• It automates documentation.

**38.Difference between priority and severity ?**

**Ans.**

|  |  |
| --- | --- |
| **Priority** | **Severity** |
| * Priority is a parameter to decide the order in which defects should be fixed. | * Severity is a parameter to denote the impact of a particular defect on the software. |
| * Priority means how fast defect has to be fixed. | * Severity means how severe defect is affecting the functionality. |
| |  | | --- | | * Priority is related to   scheduling to resolve the  Problem. | |  | | * Severity is related to the quality standard. |
| |  | | --- | | * Product manager decides the   Priorities of defects. | |  | | * Testing engineer decides the severity level of the defect. |
| |  | | --- | | * Its value is subjective. | |  | | * Its value is objective |
| * Its value changes from time to time. | * Its value doesn’t change from time to time |
| * Priority is of 3 types: Low, Medium, and High. | * Severity is of 5 types: Critical, Major, Moderate, Minor, and Cosmetic. |

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

**Ans.** Ans. There are various methodologies present in agile testing and those are listed below :-  
Scrum  
eXtreme Programming (XP)  
  
Below listed methodologies are used less frequently :-  
Dynamic System Development Method (DSDM)  
Test Driven Development (TDD)  
Feature Driven Development  
Xbreed  
Crystal

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

**Ans.**  Simply put, authentication is the process of verifying who someone is, whereas authorization is the process of verifying what specific applications, files, and data a user has access to.

* Cross Browser and device Compatibility , Responsiveness, Integration Testing, Security, Performance Testing, Application Getting Slow, Usability Testing  
    
  Entry and Exit Points, Checking the Standards and Compliance, Firewalls, Accessibility Testing, Project Deadline, User Experience, Web Service Requests,  
    
  User Input Validation.

**41.When to used Usablity Testing?**

**Ans.** Usability Testing is a significant type of software testing technique, which is comes under the non-functional testing. It is primarily used in**user-centered interaction design** on order to check the usability or ease of using a software product.

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

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

* Record and Replay GUI testing can be done using automation tools. This is done in 2 parts. ...
* Model Based Testing A model is a graphical description of a system’s behavior. It helps us to understand and predict the system behavior. ...