**Software Testing Assignment**

**Module-2 (Manual Testing)**

Q-1. What is Error, Defect, Bug and failure?

* **Error**: A mistake in coding is called error.
* **Defect**: Error found by tester is called defect.
* **Bug**: Defect accepted by development team then it is called bug.
* **Failures**: Build does not meet the requirements then it is failure.

Q-2. 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)

Q-3. What is 7 key principles? Explain in detail?

* General testing principles:

1. Testing shows presences 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
8. **Testing shows presences of defects:**

* We test to find faults.
* As we find more defects, the probability of undiscovered defects remaining in a systemreduces.
* However testing cannot prove that there are no defects present.

**2**. **Exhaustive testing is impossible:**

* Testing everything including all combinations of inputs and preconditions is not possible.
* **Example:** **Approximate total for exhaustive testing**

**20 x 4 x 3 x 10 x 2 x 100 = 480,000 tests**

**Test length = 1 sec then test duration = 17.7 days**

**Test length = 10 sec then test duration = 34 weeks**

**Test length = 1 min then test duration = 4 year.**

**Test length = 10 min then test duration = 40 year.**

**3**. **Early testing:**

* Testing activities should starts 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:**

* 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.
* Different kinds of sites are tested differently.
* For example
* Safety – critical software is tested differently from an e-commerce site.

**7**. **Absence of errors fallacy:**

* If the system built is unusable and does not fulfill the users need 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 fulfill the users needs and expectations.

Q-4. What is Exploratory Testing?

* Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking.Automation has its limits.
* Is structured and rigorous.
* Is cognitively (thinking) structured as compared to procedural structure comes from charter, time boxing etc.
* Informal testing is experience based testing.

Q-5. What is traceability matrix?

* Traceability Matrix (also known as Requirement Traceability Matrix - RTM) is a table which is used to trace the requirements during the Software development life Cycle. It can be used for forward tracing (i.e. from Requirements to Design or Coding) or backward (i.e. from Coding to Requirements)
* Types of traceability matrix

1. Forward traceability
2. Backward traceability
3. Bi-directional traceability

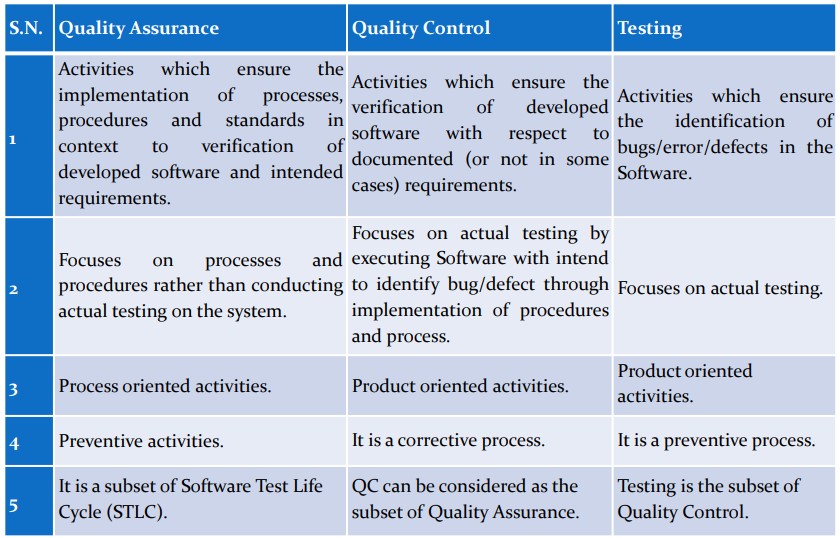
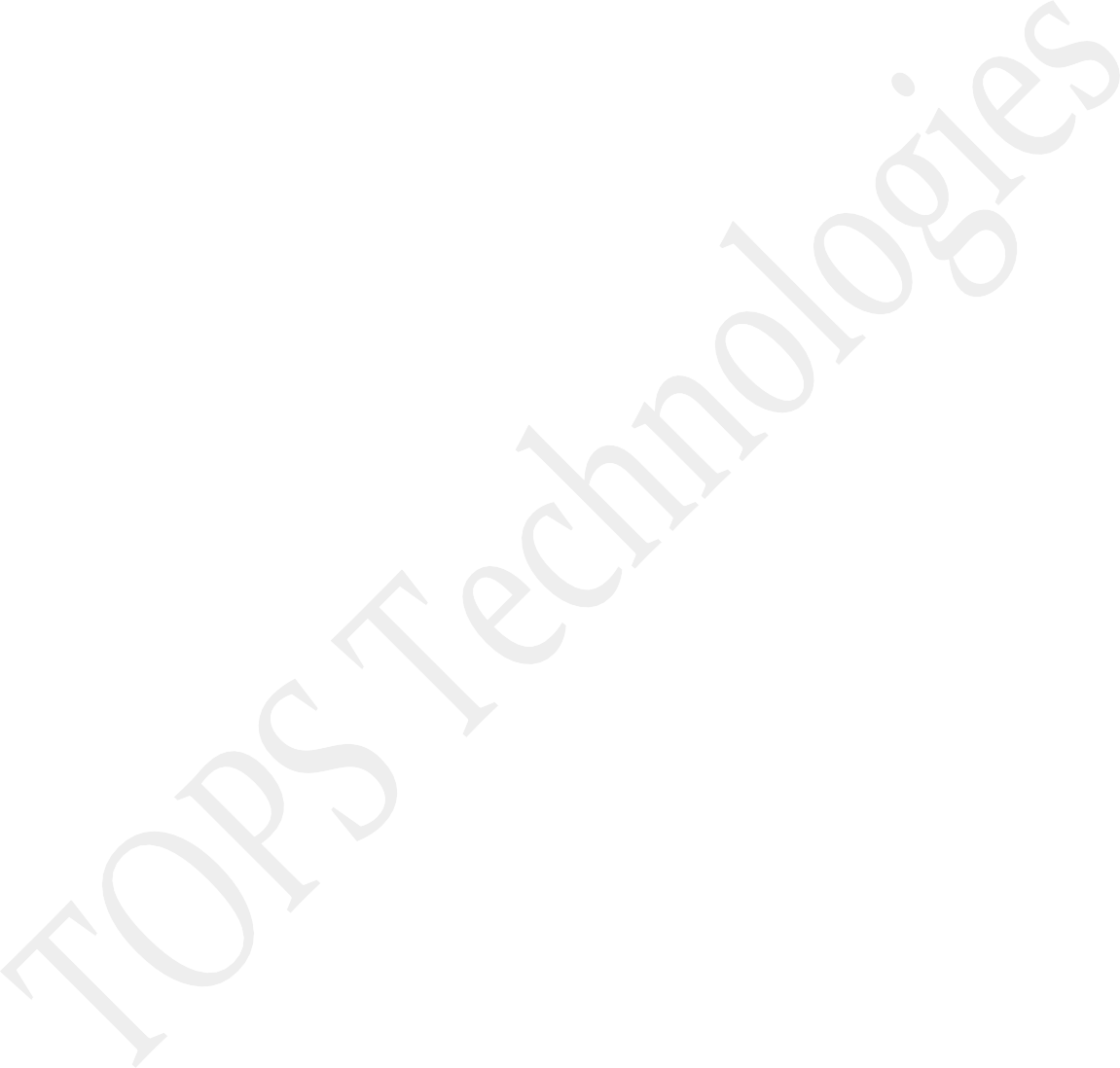
Q-6. What is Boundary value testing?

* Software testing technique in which tests are designed to include representatives of boundary values. It is performed by the QA testing teams.
* Boundary value analysis is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges.
* Boundary value analysis is a method which refines equivalence partitioning.
* Boundary value analysis generates test cases that highlights errors better then equivalence partitioning.

Q-7. 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.

Q-8. Difference between QA v/s QC v/s Tester?



Q-9. Difference between verification and Validation?

|  |  |  |
| --- | --- | --- |
| Criteria | Veriﬁcation | Validation |

|  |  |  |
| --- | --- | --- |
| Deﬁnition | The process of evaluating work-products (not the actual  ﬁnal product) of a development phas e 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. |
| Question | Are we building the product right? | Are we building the right product? |
| Evaluatio n  Items | Plans, Requirement Specs, Design Specs, Code, Test Cases | The actual product/software. |
| Activities | Reviews Walkthroughs Inspections | Testing |

Q-10. What is Integration testing?

* Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems.
* 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
* **Component integration testing**
* **System integration testing**
* **Component integration testing**: Testing performed to expose defects in the interfaces and interaction between integrated components.
* **System integration testing**: It tests the interactions between different systems and may be done after system testing.
* It verifies the proper execution of software components and proper interfacing between components within the solution.

Q-11. 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.
* The quality of systems can be improved through lessons learned from previous projects.
* Types of risk

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 1 to 10.
* **Product risk**: Example of product risks would be flight reservation system not installing in test environment.

Q-12. What is alpha testing?

* 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 form of acceptance testing.
* It comes under the category of both white box testing and black box testing.

Q-13. What is beta testing?

* 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 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.

Q-14. What is component testing?

* Component (Unit) – A minimal software item that can be tested in isolation. It means “A unit is the smallest testable part of software.”
* 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/systemare 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.
* Sometimes known as Unit Testing, Module Testing or Program Testing.
* Component can be tested in isolation – stubs/drivers may be employed.
* Unit testing frameworks, drivers, stubs and mock or fake objects are used to assist in unit testing.
* Functional and Non-Functional testing.
* Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended with debugging tool.
* It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc.

Q-15. What is functional system testing?

* This testing mainly involves black box testing and it is not concerned about the source code of the application.
* Each & every functionality of the system is tested by providing appropriate input, verifying the output and comparing the actual results with the expected results.
* This testing involves checking of User Interface, APIs, Database, security, client/ server applications and functionality of the Application under Test. The testing can be done either manually or using automation.
* Types of functional testing
* Blackbox testing
* Whitebox testing
* Experience based testing
* Smoke testing
* Sanity testing
* End to end testing

Q-16. What is Non-Functional Testing?

* Non-Functional Testing: Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability,maintainability and portability.
* May be performed at all Test levels (not just Non Functional Systems Testing)
* Measuring the characteristics of the system/software that can be quantified on a varying scale- e.g. performance test scaling
* Non-functional testing includes, but is not limited to, performance testing, load testing, stress testing, usability testing, maintainability testing, reliability testing and portability testing.
* It is the testing of “how” the system works. Non-functional testing may be performed at all test levels.
* Types of non-functional testing
* Usability testing
* Compatibility testing
* GUI testing
* Security testing
* Performance testing
* Stress testing
* Load testing

Q-17. 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.
* **Approach of GUI Testing**
* **MANUAL BASED TESTING**: Under this approach, graphical screens are checked manually by testers in conformance with the requirements stated in business requirements document.
* **RECORD AND REPLAY:** GUI testing can be done using automation tools. This is done in 2 parts. During Record , test steps are captured into the automation tool. During playback, the recorded test steps are executed on the Application under Test. Example of such tools - QTP.
* **MODEL BASED TESTING:** A model is a graphical description of system’s behavior. It helps us to understand and predict the system behavior. Models help in a generation of efficient test cases using the system requirements.

Q-18. What is Adhoc testing?

* Adhoc testing is an informal testing type with an aim to break the system.
* It does not follow any test design techniques to create test cases.
* In fact is does not create test cases altogether.
* This testing is primarily performed if the knowledge of testers in the system under test is very high.
* Main aim of this testing is to find defects by random checking.
* Adhoc testing can be achieved with the testing technique called error guessing.

Q-19. What is load testing?

* Load testing- Its a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system’s response time degrades or fails.
* Load testing is a kind of performance testing which determines a system’s performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.
* This testing usually identifies –
* The maximum operating capacity of an application.
* Determine whether current infrastructure is sufficient to run the application.
* Sustainability of application with respect to peak user load.
* Number of concurrent users that an application can support, and scalability to allow more users to access it.
* It is a type of non-functional testing. Load testing is commonly used for the

Client/Server, Web based applications – both Intranet and Internet.

Q-20. 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.
* Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions**.**
* It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions.
* Stress Testing is done to make sure that the system would not crash under crunch situations.
* Stress testing is also known as endurance testing.

Q-21. What is white box testing and list the types of white box testing?

* White box testing : 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 knownledge of how the software is implemented(code), how it work.
* 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.
* Based on code and the design of the system.
* The tests provide the ability to derive the extent of coverage of the whole application.
* White box testing techniques
* Statement coverage
* Decision coverage
* Condition coverage

Q-22. 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.
* Based on requirements.
* From the requirement, tests are created.
* Specification models can be used for systematic test case design
* Black box testing techniques
* Equivalence partitioning
* Boundary value analysis
* Decision tables
* State transition testing
* Use case testing

Q-23. Mention what bigbang 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.
* **Advantages**
* Convenient for small systems.
* **Disadvantages**
* Fault Localization is difficult.
* Given the sheer number of interfaces that need to be tested in this approach, some interfaces links to be tested could be missed easily.
* Since the integration testing can commence only after “all” the modules are designed, testing team will have less time for execution in the testing phase.

Q-24. When should "Regression Testing" be performed?

* Regression Testing: Testing of a previously tested program 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 environment is changed.
* when the system is stable and the system or the environment changes.
* when testing bug-fix releases as part of the maintenance phase.
* It should be applied at all Test Levels.
* It should be considered complete when agreed completion criteria for regression testing have been met.
* Regression test suites evolve over time and given that they are run frequently are ideal candidates for automation.

Q-25. Difference between Smoke and Sanity?

* Smoke testing

1. Check the critical functionality
2. It is done in initial stage
3. It checks the stability
4. Part of acceptance testing
5. General health check up
6. Done by tester and developer
7. It checks the system end to end
8. Twenty test cases it should take 30 min to test

* Sanity testing

1. Checks the new functionality
2. It is done after 30 build
3. It checks the sanity/rationality
4. Part of regression testing
5. Advance health check up
6. Done by tester
7. It checks only a particular function of entire system

Q-26. What is Bug Life Cycle?

* A computer bug is an error, flow, mistake, failure, or fault in a computer program that prevents it from working correctly or produces an incorrect results. Bugs arise from mistake and errors, made by people, in either a program source code or its design.
* Bug(Defect) Life Cycle

New

Assigned

---------- > Open --------------------------🡪 duplicated

rejected, deffered

Fixed not a bug

Pending reset

Reopened ←---- Retest

Verified

Closed

* Bug (defect) stages
* **New** : When a new defect is logged and posted for the first time. It is assigned a status as new.
* **Assigned** : Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team.
* **Open** : The developer starts analyzing and works on the defect fix.
* **Fixed**: When a developer make a necessary code change and verifies the change, he or she can make bug status as “fixed”.
* **Pending retest**: Once the defect is fixed the developer gives a particular code for retesting the code to the tester.
* **Retest**: Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to “re-test”.
* **Verified**: The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified.”
* **Reopen**: If the bug persists even after the developer has fixed the bug, the tester changes the status to “reopened”. Once again the bug goes through the life cycle.
* **Closed**: If the bug is no longer exists then tester assigns the status “Closed.”
* **Duplicate:** If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate.”
* **Rejected**: If the developer feels the defect is not a genuine defect then it changes the defect to “rejected.”
* **Deffered**: If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status “Deffered” is assigned to such bugs
* **Not a bug**: If it does not affect the functionality of the application then the status assigned to a bug is “Not a bug”.

Q-27. Explain types of Performance testing?

* Load testing
* Stress testing
* Endurance testing
* Spike testing
* Volume testing
* Scalability testing
* Performance problems
* Load a long time
* Load time is normally the initial time it takes an application to start
* This should generally be kept to a minimum
* Poor response time
* Response time is the time it takes from when a user input data into the application until the application outputs a response to that input
* Performance testing examples
* Web based testing & desktop-based testing
* Check the page on slow connections
* Check data-based query time
* Mobile based testing
* Check data-based query time
* Check CPU and memory usage under peak load conditions

Q-28. Mention what are the categories of defects?

* **Data Quality/Database Defects**: Deals with improper handling of data in the database.
* Examples: Values not deleted/inserted into the database properly,Improper/wrong/null values inserted in the place of the actual values.
* **Critical Functionality Defects**: The occurrence of these bugs hampers the crucial functionality of the application.
* Examples: Exceptions
* **Functionality Defects**: These defects affect the functionality of the application.
* Examples: Buttons like Save, Delete, and 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**: Application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix.
* Examples:**Authentication**:Accepting an invalid username/password

**Authorization**:Accessibility to pages though permission not given

* **User Interface Defects**: As the name suggests, the bugs deal with problems related to UI are usually considered less severe.
* Examples: Improper error/warning/UI messages

Spelling mistakes

Alignment problems

Q-29. Explain the difference between Functional testing and Non Functional testing?

* **Functional testing**
* Functional testing is executed first.
* Manual and automation testing tools can Be used for functional testing.
* Functional testing describes what the product does
* Easy to do manual testing.
* Concentrates on user requirement.
* Business requirement are the input to functional testing.
* Type of functional testing .
* **Unit testing**
* **Smoke testing**
* **Sanity testing**
* **Intergration testing**
* **White box testing**
* **Black box testing**
* **User acceptance testing**
* **Regression testing**
* **End to end testing**
* **Experience-based testing**
* **Non functional testing**
* Non-functional testing should be performed after functional testing.
* Using tools will be effective for this testing.
* Non-functional testing describes how software work.
* Tough to do manual testing.
* Concentrates on user expectation.
* Performance parameter like speed, scalability are input to non-functional testing.
* Type of non-functional testing.
* **Performance testing**
* **Load testing**
* **Volume testing**
* **Stress testing**
* **Security testing**
* **Installation testing**
* **Compatibility testing**
* **Migration testing**

Q-30. What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?

| SDLC (software development life cycle) | STLC (software testing life cycle) |
| --- | --- |
| SDLC is mainly related to software development. | STLC is mainly related to software testing. |
| Besides development other phases like testing is also included. | It focuses only on testing the software. |
| On both the development and testing process | On only testing process |
| In SDLC, more number of members (developers) are required for the whole process. | In STLC, less number of members (testers) are needed. |
| In SDLC, the development team makes the plans and designs based on the requirements. | In STLC, testing team(Test Lead or Test Architect) makes the plans and designs. |
| A good quality software product | A bug free software |
| It helps in developing good quality software. | It helps in making the software defects free. |
| SDLC phases are completed before the STLC phases. | STLC phases are performed after SDLC phases. |
| Development life cycle | Testing life cycle |

Q-31. Explain what Test Plan is? What is the information that should be covered.

* A test plan is document that describe the test scope, test strategy, objectives, deliverables and resources requirement.
* the test plan strategy along with efforts and cost estimates for the project.
* Moreover, the resources, test environment, test limitations and the testing schedule are also determined.
* The Test Plan gets prepared and finalized in the same phase.
* Below information should be covered
* Overview
* Scope
* Inclusions
* Test environment
* Exclusions
* Test strategy
* Defect reporting procedure
* Roles/responsibilities
* Test schedule
* Pricing
* Tools
* Approvals
* Risk and mitigations
* Test deliverables
* Test tools selections
* Test formations
* Test effort estimation

Q-32. What is priority?

* Priority describes the importance of the defect
* Defect priority status is the order in which defect should be fixed
* 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.
* The priority status is set based on the customer’s requirements.
* Priority can be the following types
* **Po (High)**: The defect must be resolved as soon as possible because the defect is affecting the application or the product severely. The system cannot be used until the repair has been done.
* **P1(Medium)**: The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created.
* **P2(Low)**: The defect is an irritant that should be repaired, but repair can be deferred until after a more serious defect has been fixed.

Q-33. What is severity?

* Severity describes the seriousness of defect and how much impact on business work flow.
* Severity is absolute and Customer-Focused. It is the extent to which the defect can affect the software.
* 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.
* Severity can be of the following types:
* **Critical**: The defect that results in the termination of the complete system or one or more components of the system and causes extensive corruption of the data. The failed function is unusable and there is no acceptable alternative method to achieve the required results then the severity will be stated as critical.
* **Major (High)**: The defect that results in the termination of the complete system or one or more components of the system and causes extensive corruption of the data. The failed function is unusable but an acceptable alternative method exists to achieve the required results then the severity will be stated as major.
* **Moderate (Medium)**: The defect that does not result in the termination, but causes the system to produce incorrect, incomplete or inconsistent results then the severity will be stated as moderate.
* **Minor (Low)**: The defect that does not result in the termination and does not damage the usability of the system and the desired results can be easily obtained by working around the defects then the severity is stated as minor.

Q-34. Explain the difference between Authorization and Authentication in Web testing. What are the common problems faced in Web testing?

* Authentication: Accepting an invalid username/password.
* Authorization: Accessibility to pages though permission not given.

Q-35. What is bug categories?

* **Duplicate:** If the defect is repeated twice or the defect corresponds to the same concept of the bug it called duplicate.
* **Rejected**: If the developer feels the defect is not a genuine defect then it changes the defect to “rejected.”
* **Deffered**: If the present bug is not of a prime priority and if it is expected to get fixed in the next release it called deffered.
* **Not a bug**: If it does not affect the functionality of the application then it called not a bug.