Manual Interview Questions and Answers

1. What is manual testing?

Ans: Manual Testing is a type of software testing in which test cases are executed manually by a tester without using any automated tools. The purpose of Manual Testing is to identifythe bugs, issues, and defects in the software application. Manual software testing is the most primitive technique of all testing types and it helps to find critical bugs in the software application. Any new application must be manually tested before its testing can be automated.

2. Black box testing and white box testing?

Ans: It is an Interface testing by Test Engineer. Internal system testing is not considered in this type of testing. Tests are based on customer requirements and Functionality. Other than Unit Level testing, the remaining testing types belongs to Black box testing.

The White Box Test method is the one that looks at the code and structure of the product to be tested and uses that knowledge to perform the tests. This method is used in the Unit Testing phase, although it can also occur in other stages such as Integration Tests. For the execution of this method, the tester or the person who will use this method must have extensive knowledge of the technology used to develop the program.

3. What is data base testing?

Ans: Database Testing is a type of software testing that checks the schema, tables, triggers etc. of the database under test. It involves creating complex queries for performing the load or stress test on the database and check its responsiveness. It checks integrity and consistency of data. Database testing is important for ensuring that the data in the database is accurate, consistent, and secure. The database is a critical component of any software application, and errors in the database can result in serious consequences for the organization, including data loss, data corruption, and security breaches.

Some of the commonly used techniques for database testing include data integrity testing, performance testing, security testing, and data migration testing. Testing tools and frameworks

such as SQL Server Management Studio, MySQL Workbench, and Apache JMeter are often used for database testing.

4. Software testing types?

Ans: Accessibility testing **Acceptance testing** **Black box testing**

End to end testing Functional testing Interactive testing
Integration testing Load testing Non-functional testing
Performance testing Regression testing Sanity testing
Security testing Single user Performance testing
Smoke testing Stress testing Unit testing
White-box testing

5. What is Test cases?

Ans: A test case is a document which contains set of input values, execution preconditions, expected results and developed for a particular objective, such as to verify compliance(fulfilment) with a specific requirement. Test cases define how to test a system, software or an application. A test case is a singular set of actions or instructions for a tester to perform that validates a specific aspect of a product or application functionality.

If the test fails, the result might be a software defect that the organization can triage.

6. Defect life cycle?

Ans: Defect cycle or defect life cycle is ride of a defect from discovering defect to closure of defect. Defects management in defect cycle is important to ensure the software quality. Preventing, identifying, rectifying defect is important to improve the quality.

Defect Life Cycle States:

New - Potential defect that is raised and yet to be validated.

Assigned - Assigned against a development team to address it but not yet resolved.

Active - The Defect is being addressed by the developer and investigation is under progress. At this stage there are two possible outcomes; viz - Deferred or Rejected.

Test - The Defect is fixed and ready for testing.

Verified - The Defect that is retested and the test has been verified by QA.

Closed - The final state of the defect that can be closed after the QA retesting or can be closed if the defect is duplicate or considered as NOT a defect.

Reopened - When the defect is NOT fixed, QA reopens/reactivates the defect.
Deferred - When a defect cannot be addressed in that particular cycle it is deferred to future release.

Rejected - A defect can be rejected for any of the 3 reasons; viz - duplicate defect, NOT a Defect, Non-Reproducible.

7. Tell me about your project?

Ans: My Project name is "ERPPMC". It has different modules and each module having different pages.

Modules are:

1. Customer Module,
2. Purchase Module,
3. Sales Module,
4. Vendor Module,
5. Inventory Module and Finance Module.

8. What is difference between manual and automation?

Ans: There are some major differences between manual testing and automation testing. In manual testing, a human performs the tests step by step, without test scripts. In automated testing, tests are executed automatically via test automation frameworks, along with other tools and software.

9. Why you choosing manual testing?

Ans: Any software or product is incomplete without its testing. Without testing, there could be some bugs in the product and customers will not get satisfied according to the requirements. For testing, all companies need to have software tester for complete and satisfied software or product without any errors or bugs.

10. Smoke (or) Quick Testing and Sanity Testing (or) Build Test?

Ans: In smoke testing verify the build readiness for testing.

Here we are verifying whether build is ready for testing or not by testing all major functionalities (with valid data and without test cases).

→Sanity testing is a software testing technique in which a particular functionality of the application is verified instead of carrying out regression on the complete application. In this way, we can say that sanity testing is a subset of regression testing. Just like regression testing, in the case of sanity testing, we check if a fix has not affected the previously working functionalities of the application. But in lesser time and with a lesser number of test cases, focusing on some critical parts only.

11. What is block box testing?

Ans: It is an Interface testing by Test Engineer. Internal system testing is not considered in this type of testing. Tests are based on customer requirements and Functionality. Other than Unit Level testing, the remaining testing types belongs to Black box testing.

Black box testing Two types:

1.Functional Testing

2.Non-functional Testing

12. What is Regression Testing?

Ans: It is Re-execution of some are all test cases of a testing activities for each build to verify that fixes or changes made have not introduced new defects.

Regression testing is done in three situations.

1. After fixing the bug.

2. New change request will come from client.

3. When environment changes.

Here we have to verify whether already existing functionality can get defects or not.

13. Why we choose testing?

Ans: One company is developing any application the company wants to know whether the

developed application is working as per our requirements or not. So immediately to test the application. Testing is the process of verifying the defects of the developed application.

14. SDLC Process?

Ans: SDLC Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software's. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life Cycle has its own process and deliverables that feed into the next phase.

15. A tester has found a defect where the defect is important to such that it has to be fixed the current release select the priority of the from the below options.

Ans. P1 Priority

16. Positive testing and negative testing?

Ans: **Positive Testing** is a type of testing which is performed on a software application by providing the valid data sets as an input. It checks whether the software application behaves as expected with positive inputs or not. Positive testing is performed in order to check whether the software application does exactly what it is expected to do.

Negative Testing is a testing method performed on the software application by providing invalid or improper data sets as input. It checks whether the software application behaves as expected with the negative or unwanted user inputs. The purpose of negative testing is to ensure that the software application does not crash and remains stable with invalid data inputs.

17. What is latent defect?

Ans: Defects that are not identified during Testing phase and are later identified after the software has been released into the market by real users using a specific scenario and specific data set. Latent defect is a term used in software engineering and quality assurance to refer to a defect or problem in the software application that is not immediately apparent or visible. It is a defect that remains hidden or dormant until a particular set of conditions are met or until the application is used in a specific way.

18. What is test case design techniques?

Ans: Test case design techniques are broadly divided into two types.

1. Black box testing (Functional).
2. White box testing (Structural).
   Black box Test case design techniques
   1. Equivalence Class Partitioning (ECP).
   2. Boundary Value Analysis (BVA).
   3. State transition testing.
   4. Cause-effect testing.

19. What is Data Testing?

Ans: In testing process, we need test data for validating the application. In this process, checking the prepared test data is suitable for testing the application or not is called Data Testing. Data testing is a type of software testing that is performed to verify that the data in the software application is accurate, complete, and consistent. It is an important aspect of software testing as data is a critical component of any software application, and errors or discrepancies in the data can have significant consequences for the organization.

Some of the commonly used techniques for data testing include data integrity testing, performance testing, security testing, and data migration testing. Data testing can also involve the use of test data management tools to generate test data and simulate different types of user input.

20. What is a Bug?

Ans: Problem which is identified on Test Engineer machine i.e called Bug (or) Defect. Expected result is not matching with the application actual result is called Bug. Any flaw (or) imperfection in software work product (Deliverable).

Bugs can be categorized into different types based on their impact and severity, such as critical bugs, major bugs, minor bugs, and cosmetic bugs. Critical bugs are the most severe type of bugs and can result in data loss, system crashes, or other serious issues. Major bugs can cause significant functionality issues or system errors, while minor bugs can cause minor issues that do not significantly impact the user experience. Cosmetic bugs, also known as visual bugs, are minor bugs that only affect the appearance of the application and do not impact its functionality. Effective bug management is a critical component of software development and involves the use of bug tracking tools to identify, report, and track bugs throughout the

development process. Proper bug management helps to ensure that bugs are quickly identified and resolved, resulting in a more stable and reliable software application.

21. What is compatibility testing?

Ans: Compatibility Testing is a type of Software testing to check whether your software is capable of running on different hardware, operating systems, applications, browsers, versions, network environments or Mobile devices.

Hardware: It checks software to be compatible with different hardware configurations.

Operating Systems: It checks your software to be compatible with different Operating Systems like Windows, Unix, Mac OS etc.

Software: It checks your developed software to be compatible with other software. For example, MS Word application should be compatible with other software like MS Outlook, MS Excel, VBA etc.

Network: Evaluation of performance of a system in a network with varying parameters such as Bandwidth, Operating speed, Capacity. It also checks application in different networks with all parameters mentioned earlier.

Browser: It checks the compatibility of your website with different browsers like Firefox, Google Chrome, Internet Explorer etc.

Devices: It checks compatibility of your software with different devices like USB port Devices, Printers and Scanners, Other media devices and Bluetooth.

Mobile: Checking your software is compatible with mobile platforms like Android, iOS etc.

Versions of the software: It is verifying your software application to be compatible with different versions of the software. For instance, checking your Microsoft Word to be compatible with Windows 7, Windows 7 SP1, Windows 7 SP2, Windows 7 SP3.

22. What is STLC?

Ans: STLC stands for Software Testing Life Cycle. STLC is a process used to test software and ensure that quality standards are met. The STLC is a process that follows a sequential set

of steps that start with planning and end with closure. The main objective of STLC is to ensure that the software product is of high quality, meets the customer requirements, and is delivered on time and within budget. Tests are carried out systematically over several phases.

STLC Phases:

1. Requirement Analysis.
2. Test Planning.
3. Test Design.
4. Test Environment Setup.
5. Test Execution.
6. Defect Logging.
7. Defect Retesting
8. Test Closure.

23. Severity, priority with examples?

Ans: **Severity** defines the importance of defect with respective to functional point of view…. means criticalness of defect with respective to application.

Responsibility of "Testing Team".

Classification could be:

*CRITICAL *MAJOR *MODERATE *MINIMAL

Priority defines the importance of defect with respective to client point of view… means how soon it should fix.

Responsibility of developing team (DL, Tech Lead)

Classification could be:

*P1(Urgent) *P2(High) *P3(Medium) *P4(Low)

24. What is SDLC Principles?

Ans: Implementing a SDLC is all about quality, reducing costs and saving time. Embracing the SDLC principles will improve your quality assurance practices, increase your project

success rate, reduce rework and provide deliverables that meet or exceed your stakeholders' expectations.

Principle #1: An effective organizational change management strategy is essential to the success of the SDLC.

Principle #2: Investment opportunities cannot be estimated accurately until the requirements are known.

Principle #3: Build systems with minimal defects.

Principle #4: Always conduct acceptance testing.

Principle #5: Complete all project artifacts before implementation.

25. Test design techniques?

Ans: Test design techniques are methods used to create a set of effective and efficient test cases that will help in achieving maximum test coverage. These techniques are designed to help testers create comprehensive test cases, optimize test coverage, and minimize the number of test cases required to achieve thorough testing. Some of the commonly used test design techniques include:

1. **Equivalence Partitioning:** This technique involves dividing the input data into different classes or partitions based on similar characteristics, and then selecting test cases from each partition.
2. **Boundary Value Analysis:** This technique involves selecting test cases that focus on the boundary values of the input data, as they are more likely to reveal defects than other values.
3. **Decision Table Testing:** This technique involves creating a table that lists all possible combinations of inputs and expected outputs, and then selecting test cases from this table.
4. **State Transition Testing:** This technique is used for testing systems that have different states or modes, and involves selecting test cases that exercise the transition between different states.
5. **Exploratory Testing:** This technique involves testing the application in an unscripted

manner, where the tester explores the application and identifies defects that may not be discovered by other techniques.

6. **Error Guessing:** This technique involves using the tester's experience and intuition to identify areas of the application that may be more error-prone, and then selecting test cases to exercise these areas.

26. Which is better among manual and automation Testing?

Ans: Manual testing and automation testing both have their own advantages and disadvantages, and the choice between the two depends on various factors, such as the project requirements, timelines, budget, and the expertise of the testing team. In some cases, a combination of both manual and automation testing may be necessary to achieve maximum test coverage. A skilled testing team should be able to evaluate the requirements of the project and determine the best approach to achieve the desired results.

Manual testing involves the execution of test cases manually by a tester without the use of any automated tools. Manual testing is generally recommended for small projects or when the test cases require frequent updates, as manual testing allows testers to quickly adapt to changes in the requirements. Manual testing is also beneficial when it comes to exploratory testing, where the tester can use their expertise and intuition to identify defects in the software.

Automation testing involves the use of automated testing tools to execute test cases. Automation testing is recommended for large projects or when the test cases require repetition, as it saves time and effort by automating the testing process. Automated testing is also beneficial for testing large and complex systems where manual testing can be time-consuming and error-prone.

27. How to identify the bugs?

Ans: As a software tester, identifying bugs involves several steps, including:

Test case execution: Run a set of test cases and compare the expected results with the actual results to identify any discrepancies.

Review of requirements: Compare the requirements to the actual product to identify any discrepancies or missing functionality.

Reproduce the issue: Try to recreate the issue multiple times to make sure it's a bug and not a one-time issue.

Gather information: Collect information about the environment, steps to reproduce the issue, and any error messages or logs.

Defect logging: Document any issues found during testing by creating a defect report, including steps to reproduce and expected vs actual results.

Exploratory testing: Ad-hoc testing that aims to uncover issues by randomly trying out different scenarios and combinations of inputs.

Logs and error messages: Review logs and error messages to identify any issues or exceptions thrown by the system.

User feedback: Collect feedback from end-users to identify any issues or pain points with the product.

Regression testing: Re-run previously passing tests to ensure new changes have not introduced new bugs.

Test automation: Use automated testing tools and scripts to identify bugs in a systematic and consistent manner.

28. Explain about critical issue faced in your defects?

Ans: A defect that has completely blocked the functionality of an application where the user or the tester cannot proceed or test anything. If the whole application's functionality is inaccessible or down because of a defect, such a defect is categorized as a critical defect. Critical issues are typically considered high-priority and require immediate attention and resolution. A critical defect is a show stopper which means the functionality cannot be delivered unless it is fixed. A critical defect has to be fixed immediately and ASAP since it can block testing of an entire product.

⇒ User cannot log in to the account.

⇒ A feature developed is missing in the build 'or' complete failure of a feature.

⇒ Internal Server error on accessing the application.

⇒ Error while customer is making the payment in case of eCommerce site, as the business loses its money.

⇒ Incorrect calculation or reporting of critical business data

⇒ Failures in key functionality, such as a shopping cart not calculating total cost correctly.

29. Which of the following is not a non-functional testing?

Ans: Here is a list of common types of non-functional testing:

Performance testing: Testing the system's performance, response time, and scalability under different loads and conditions.

Load testing: Testing the system's ability to handle expected levels of traffic and concurrent users.

Stress testing: Testing the system's behavior and stability under extreme loads and conditions.

Endurance testing: Testing the system's behavior and performance over an extended period of time.

Scalability testing: Testing the system's ability to accommodate increased traffic and load.

Compatibility testing: Testing the system's compatibility with different platforms, devices, browsers, and configurations.

Usability testing: Testing the ease of use and user experience of the system.

Security testing: Testing the system's security features and protections against potential threats and attacks.

Recovery testing: Testing the system's ability to recover from failures and errors, and return to normal operation.

Localization testing: Testing the system's behavior and display of content in different languages and cultural environments.

30. What is vulnerability testing?

Ans: Vulnerability testing is a type of security assessment that identifies and evaluates the security weaknesses or vulnerabilities in a system, network, or application. The purpose of

vulnerability testing is reducing the possibility for intruders/hackers to get unauthorized access of systems. Vulnerability testing is important because it helps organizations to proactively identify and address security risks, rather than waiting for an attack to occur. This can help to prevent data breaches, loss of confidential information, and other security incidents that can have serious consequences for the organization and its customers. It is performed to identify potential security risks and to evaluate the effectiveness of security controls in place.

31. What is Alpha Testing and Beta Testing?

Ans: Alpha Testing: Testing of the whole computer system before rolling out to the UAT. Alpha testing is carried out by the testers who are internal employees of the organization. Alpha testing performed by Testers who are usually internal employees of the organization.

Deliverable: Stable application

32.Differences between Smoke and Sanity, Retesting and Regression testing?

Smoke Testing	**Sanity Testing**	**Regression Testing**	**Re-testing**
performed before any detailed testing and is used as a quick check to determine if the application is stable enough to proceed with further testing.	used to ensure that the application is still functioning as expected after a minor change.	performed after changes have been made to the code and is used to ensure that the application is still functioning as expected after these changes.	bug has been completely fixed and that the application is functioning as expected.
Performed on initial builds	Performed on stable builds	Performed on stable builds	Performed on stable builds
Executed by testers & sometimes also by developers	Executed by testers	Executed by testers, mostly via automation	Executed by testers

Defect verification is not the part of smoke testing	Defect verification is not the part of sanity testing	Defect verification is not the part of regression testing	Defect verification is the part of re-testing
The purpose of smoke testing is to verify the "stability" of the system in order to proceed with more rigorous testing	The purpose of sanity testing is to verify the "rationality" of the system in order to proceed with more rigorous testing	The purpose of regression testing is that new changes should not have any side effects to existing functionalities	Re-testing is done on the basis of the defect fixes

33. Write test cases for email id?

Ans: Design or UI related Test Cases:

- Verify email field is present on the page.
- Verify label text is shown with the email field or not.
- Verify label text email align with the email field.
- Verify that the placeholder text in the email field is added or not.
  Functional Test Cases:
- Verify email address field is accessible by clicking on the email field.
- Check users can type email in the email field.
- Verify user can paste the email address in the field by keyboard keys Ctrl + v.
- Verify that the user can paste the email address with the mouse by right-clicking in the email field and pasting the email address.
- Verify validation for the email field is implemented or not.
- Verify an error message should be shown in case if the user adds an invalid email address or not.

  Here are some test cases for validating an email ID:

1. Test for a valid email ID format:
   Input: user@example.com
   Output: Email ID is valid

2. Test for an invalid email ID format (missing @ symbol):

   Input: userexample.com

   Output: Email ID is invalid

3. Test for an invalid email ID format (missing domain name):

   Input: user@.com

   Output: Email ID is invalid

4. Test for an invalid email ID format (missing user name):

   Input: @example.com

   Output: Email ID is invalid

5. Test for an invalid email ID format (missing. in domain name):

   Input: user@examplecom

   Output: Email ID is invalid

6. Test for an invalid email ID format (using spaces):

   Input: user @example.com

   Output: Email ID is invalid

7. Test for an invalid email ID format (using special characters other than . and @):

   Input: user#example.com

   Output: Email ID is invalid

8. Test for a maximum length email ID:

   Input:user12345678901234567890123456789012345678901234567890123@example.com

   Output: Email ID is valid

9. Test for an email ID with multiple dots in the domain name:

   Input: user@example.com.au

   Output: Email ID is valid

34. How you start testing process?

Ans: Starting the testing process involves the following steps:

1. Requirements gathering: Gather the requirements and specifications of the software to be tested. This will help to understand what needs to be tested and what the expected outcome should be.

2. Test planning: Create a test plan that outlines the scope, objectives, and approach for testing. The test plan should also include the resources, timeline, and budget for testing.
3. Test case development: Develop test cases based on the requirements and test plan. Test cases should include steps for testing, expected results, and any additional information needed to execute the tests.
4. Test environment setup: Set up the test environment, including hardware, software, and any necessary tools. This may involve installing the software, configuring the test environment, and preparing test data.
5. Test execution: Execute the test cases and document the results. This may involve manual testing, automated testing, or a combination of both.
6. Defect tracking: Track any defects that are found during testing and report them to the development team for resolution.
7. Test closure: Close the testing phase when all the tests have been executed and the results have been documented. This may involve preparing a final test report and documenting any lessons learned during the testing process.

35. What is test coverage?

Ans: Test coverage is defined as a statistic that indicates the quantity of testing completed by a collection of tests. Test coverage is defined as a metric in Software Testing that measures the amount of testing performed by a set of tests. It will include gathering information about which parts of a program are executed when running the test suite to determine which branches of conditional statements have been taken.

Test coverage (TC)=Test cases executed / total number of Test cases *100

36. what is Exhaustive testing?

Ans: Exhaustive testing is a testing method in which all possible combinations of inputs, conditions, and scenarios are tested. The goal of exhaustive testing is to ensure that all possible scenarios have been considered and that no defects or bugs exist in the software. Exhaustive testing is sometimes known as ‘complete testing’ and aims to reduce the chance of undiscovered faults in a software package. A team of software ‘testers’ will test as many possible input combinations and conditions, looking for bugs and glitches. Exhaustive testing is typically used in situations where high reliability and safety are required, such as in the

aerospace, defense, and medical industries. In most other cases, exhaustive testing is not practical or necessary, and other testing methods, such as risk-based testing or scenario-based testing, are used instead.

Exhaustive testing should be used in combination with other testing methods, such as unit testing, integration testing, and system testing, to provide a comprehensive testing approach.

37. What is Design?

Ans: Design refers to the process of planning and creating a solution to meet a particular need or requirement. It involves visualizing, conceptualizing, and defining the structure, form, and function of a product or system.

Design in software testing refers to the process of creating a plan or blueprint for testing a software application. It involves determining what tests need to be performed, what test cases will be used, and what testing tools and techniques will be employed.

1. The software requirements: The testing plan should ensure that all the requirements of the software are covered in the testing process.
2. The software architecture: The testing plan should take into account the software architecture and how it will affect the testing process.
3. The software environment: The testing plan should consider the software environment, including the hardware and software configurations, network connections, and data sources.
4. The testing goals: The testing plan should clearly define the goals of the testing process and how they will be achieved.
5. The testing schedule: The testing plan should include a schedule for when each test will be performed, who will perform it, and what resources will be required.

38.100 cases are there, then how these cases are executed?

Ans: The best approach to executing test cases will depend on the specific needs and constraints of the project. It may be necessary to use a combination of these approaches to ensure that all the test cases are executed effectively and efficiently.

1. Test prioritization: Prioritizing the test cases based on their criticality and risk, and executing the high-priority test cases first, can help to focus the testing efforts on the

most important issues.

2. Test parallelization: Running multiple tests at the same time on different testing environments can help to reduce the overall testing time.
3. Test optimization: Optimizing the test cases by reducing the number of test steps, reducing the time required to set up the testing environment, and reusing test data can help to reduce the time required to execute the tests.
4. Test environment management: Properly managing the testing environment, such as ensuring that it is stable, well-configured, and up-to-date, can help to reduce the time required to execute the tests.
5. Test data management: Using optimized and well-managed test data can help to reduce the time required to execute the tests, as well as increase the consistency and accuracy of the results.
6. Manual execution: In this approach, the test cases are executed manually by a tester, who follows the steps outlined in the test cases and records the results. This is often done for smaller projects or when a high degree of human judgment is required.
7. Automated execution: In this approach, test cases are executed using automated testing tools, such as test scripts or test frameworks. This can help to reduce the time and effort required to execute the tests, as well as increase the consistency and accuracy of the results.

By using these strategies, you can significantly reduce the time required to execute 100 test cases and ensure that the tests are executed efficiently and effectively.

39. How do you experienced your test?

Ans: The experience of software testing can vary depending on the specific software being tested, the testing techniques being used, and the goals of the testing process. In general, software testing can be a time-consuming and meticulous process, but it is an essential step in the software development cycle to ensure the quality and reliability of the final product.

In software testing, the goal is to validate and verify that a software application meets its specified requirements and works as intended. This is usually done through a combination of manual and automated testing techniques. The software is subjected to various tests, such as functional testing, performance testing, security testing, etc. The results of these tests are then

analyzed to identify any bugs or issues that need to be fixed.

40. Testing techniques?

1. Manual testing – Involves manual inspection and testing of the software by a human tester.
2. Automated testing – Involves using software tools to automate the testing process.
3. Unit Testing: It is a method of testing individual units or components of a software application.
4. Integration Testing: It involves testing the interactions between different components or systems to ensure they work together as expected.
5. Functional Testing: This type of testing verifies that the software functions according to the specified requirements.
6. End-to-end Testing: It tests the complete software system from start to finish to ensure that all the components work together seamlessly.
7. Regression Testing: It is performed to verify that changes made to the software have notintroduced new bugs or affected existing functionality.
8. Performance Testing: This type of testing is performed to evaluate the speed, stability, and scalability of the software under different conditions.
9. Security Testing: It is a type of testing that is performed to evaluate the security of the software and identify any vulnerabilities.
10. User Acceptance Testing (UAT): This type of testing is performed by the end-users of the software to verify that it meets their requirements and expectations.

41---- is to check the coverage of requirements in the test design?

Ans: Testing coverage of requirements is an important part of the software development process because it helps identify any gaps or missing requirements and ensures that the software is of high quality and meets the expectations of the end-users.

Yes, checking the coverage of requirements in the test design is an important goal of software testing. The aim is to ensure that all relevant requirements, both functional and non-functional, have been included in the test design and are being adequately tested.

To achieve this goal, testers often create a test plan that outlines the requirements that need to be tested and the methods that will be used to test them. The test plan serves as a roadmap for

the testing process and helps ensure that all of the requirements are thoroughly tested.

42. What is bug life cycle?

Ans: The Defect Life Cycle, also known as the Bug Life Cycle, is a cycle of defects from which it goes through covering the different states in its entire life. This starts as soon as any new defect is found by a tester and comes to an end when a tester closes that defect assuring that it won't get reproduced again.

1. Detection: A bug is identified and detected during testing.
2. Report: The bug is reported and documented in a bug tracking system.
3. Prioritization: The bug is prioritized based on its severity and impact on the system.
4. Investigation: The bug is investigated by the development team to determine its root cause.
5. Fix: A solution is developed to resolve the bug.
6. Verification: The solution is tested to verify that the bug has been fixed.
7. Closure: The bug is marked as resolved and closed in the bug tracking system.

43.Verification &Validation?

Ans: Verification in software testing refers to the process of evaluating the software design and implementation to determine if they meet the specified requirements. It is a process of checking that the software product is designed and developed correctly and meets the quality standards. The main goal of verification is to catch errors and mistakes early in the development cycle, before they become more costly and time-consuming to fix. This process includes activities such as code review, walk-through, inspection, and testing prototypes. Verification helps ensure that the software is built to meet the user's needs and expectations.

Validation in software testing is the process of evaluating a software product during or at the end of the development process to determine whether it satisfies the specified requirements. The main objective of validation is to assess the software's compliance with the user's needs and expectations, and to ensure that it is fit for its intended use. This process includes activities such as functional testing, integration testing, system testing, and acceptance testing. The goal of validation is to identify any defects or issues that may affect the software's functionality and performance, and to ensure that the software meets the necessary quality standards before it is released to the end-user. Validation helps to reduce the risk of releasing a software product

that does not meet the customer's requirements, leading to customer dissatisfaction and increased costs for fixing the defects.

44. Write test cases on login page?

Ans: Test cases that can be performed on any login page:

Positive test cases:

1. Test the login functionality with a valid username and password combination.
2. Test the case-insensitivity of the username field.
3. Test the maximum and minimum length of the username and password fields.
4. Test the successful login with special characters in the username and password.
5. Test the "remember me" feature by checking if the login credentials are saved after logging out.

Negative test cases:

1. Test the login functionality with an invalid username and password combination.
2. Test the login functionality with an empty username and password.
3. Test the login functionality with a username that does not exist.
4. Test the login functionality with an incorrect password for a valid username.
5. Test the login functionality with an account that has been disabled or locked.
6. Test the error messages displayed for invalid login attempts.

45. Write test cases for Gmail login page?

Ans: Here are some test cases that can be performed on the Gmail login page:

Positive test cases:

1. Test the login functionality with a valid Gmail account.
2. Test the case-insensitivity of the username field.
3. Test the successful login with special characters in the username and password.
4. Test the "Stay signed in" feature by checking if the login credentials are saved after logging out.
5. Test the "Forgot password" feature and verify that the reset password process works.
6. Test the "Two-step verification" feature and verify that the process works as expected.

Negative test cases:

1. Test the login functionality with an invalid Gmail account.
2. Test the login functionality with an empty username and password.
3. Test the login functionality with an incorrect password for a valid Gmail account.
4. Test the error messages displayed for invalid login attempts.
5. Test the login functionality with an account that has been disabled or locked.

These are just some of the basic test cases that can be performed on the Gmail login page.

46. What is performance testing, types

Ans: Checking the behavior of an application by applying some load is known as performance testing. Performance testing is a non-functional testing. While doing performance testing onthe application, we will concentrate on the various factors like Response time, Load, and Stability of the application.

Types of Performance Testing:

1. Load testing
2. Stress testing
3. Scalability testing
4. Stability testing

1.Load testing: To check the performance of an application by applying some load which is either less than or equal to the desired load is known as load testing.

2.Stress Testing: The stress testing, checks the behavior of an application by applying load greater than the desired load.

3.Scalability Testing: Checking the performance of an application by increasing or decreasing the load in particular scales (no of a user) is known as scalability testing.

4.Stability Testing: Checking the performance of an application by applying the load for a particular duration of time is known as Stability Testing.

47. What is defect components?

Ans: Defect Components

1. Defect ID.

2. Defect Description.
3. Version.
4. Action Steps.
5. Expected Result.
6. Environment.
7. Actual Result.
8. Severity.
9. Detected By
10. Fixed by
11. Date Raised
12. Priority
13. Date Closed

48. WhatsApp test scenarios?

Ans: **WhatsApp Test Scenarios:**

⇒ Verify that on downloading the WhatsApp application, users can register using a new mobile number.

⇒ Verify that for a new mobile number user will get a verification code on his mobile and filling the same verifies the new user account.

⇒ Check the maximum number of incorrect attempts allowed while filling the verification code.

⇒ Verify that registering an existing mobile number for new user account registration is not allowed.

⇒ Verify that on successful registration all the contacts in the user's contact directory get imported to the WhatsApp contact list.

⇒ Verify that the user can set DP and status on WhatsApp.

⇒ Verify that the user can update the existing DP and WhatsApp status.

⇒ Verify that the user can send messages to any individual selected from his contact list.

⇒ Verify that 'Chats' window contains all the chat list with DP and name and last message preview of the other person with whom chat was initiated.

⇒ Verify that clicking a chat in the chat list opens a new window containing all the chats

received and sent with the other person.

⇒ Verify that the user can check the message delivered and read time for a message in the

⇒ 'Message Info' section.

⇒ Verify that the user can share or receive contact with the other person.

⇒ Verify that the user can create a group adding multiple people from his contact list.

⇒ Verify that the user can send and receive the message in group chats.

⇒ Verify that users can send and receive images, audio, video, emoticons in chat to individuals.

⇒ Verify that users can send and receive images, audio, video, emoticons in group chats.

⇒ Verify that the user can send and receive chats in secondary languages available.

⇒ Verify that users can delete text, images, audio, video messages within a chat.

⇒ Verify that users can clear complete chat history in an individual or group chat.

⇒ Verify that users can archive chats in an individual or group chat.

⇒ Verify that users can block a user to prevent any message from getting received from the blocked contact.

⇒ Verify that the user makes WhatsApp calls to the person in his contact list.

⇒ Verify that the user can receive WhatsApp calls from a person in his contact list.

⇒ Verify that users can mark chats as favorite and access all chats marked as favorite from the 'Favorites' section.

49. WhatsApp test cases?

Ans: Some of important test cases for whatsapp.

1. First Verify that the after successfully downloading the application user able to register mobile number or not.

2. Verify that the user able to verify the number successfully or not.

3. Verify that when user enter the mobile number in whatsapp app which is already register.

4. Verify that after the successfully verification all the element properly visible / present or not. (Contact list, menu bar, chat status, calls etc.).

5. Verify that all the whatsapp contact list showing or not.

6. Check that user able to set the dp or not.

7. Check that user able to set the status or not on whatsapp app. 8. Check that user able to send the whtasapp message to any whatspp contact.

9. Check that user able to send the message by selecting attaching the photo, video, file etc through the whatsapp app.

10. Verify that by default Chat option selected or not.

11. Verify that all the chat list showing or not .(whith last message) on chat window.
12. Verify that on clicking on status window, Status of contact sowing or not.

13. Check that on clicking on calls window all the incoming and outgoing audio video call list showing or not.

14. Check that when user click on gallery option it redirects user to the phone gallery or not.

15. Check that user able to select the image from the gallery or not.

16. Check that after the selection of picture resize option showing or not.

17. Check that user able to resize the image or not.

18. Check that when user click on done option then selected image get updated as dp or not and confirmation message showing or not.

19. Check that when user click on cancel button then it redirects user to in gallery or not.

20. Check that when user select the camera option then mobile phone camera get on or not.

50. Difference between Test design and Plan?

Ans: Test design focuses on the creation of test cases, while the test plan focuses on the overall strategy and plan for testing the software application. The test design provides the details for executing the tests, while the test plan provides the overall framework and direction for the testing effort.

Test design is the process of creating and documenting the test cases that will be used to validate the functionality of a software application. The goal of test design is to ensure that the software is tested thoroughly and that all potential issues are identified and addressed. Test design involves identifying the requirements, defining the test objectives, developing the test

cases, and choosing the appropriate testing methods.

Test plan, on the other hand, is a document that outlines the approach, resources, and schedule for testing a software application. It provides a high-level overview of the testing process and includes information such as the scope of testing, the testing methodologies to be used, the resources required, the schedule for testing, and the criteria for success. The test plan is a livingdocument that is updated throughout the testing process as changes are made to the software application or the testing approach

51. Write Test case template?

Ans: Here is a sample test case template that you can use:

Test Case ID: [Unique identifier for the test case]

Test Case Title: [Brief description of the test case]

Test Designed by: [Tester's Name]

Date of test designed: [Date when test was designed]

Test Executed by: [Who executed the test- tester]

Date of the Test Execution: [Date when test needs to be executed]

Test Steps: [Mention all the test steps in detail and write in the order in which it requires to be executed. While writing test steps ensure that you provide as much detail as you can]

Test Data: [List of any necessary data required to execute the test case, if applicable]

Test Environment: [Description of the environment in which the test case will be executed, if applicable]

Expected Result: [Description of the expected result of the test case]

Actual Result: [Result of the test case, filled after execution]

Status (Fail/Pass): Mark this field as failed, if actual result is not as per the estimated result

A	B	C	D	E	F	G	H	I	J	K	L	M	N
TEST CASES TEMPLATE													
Componay Logo	Project Code							Total No of Tc'S					
	Project Name							Passed					
	Prepared By		Date Prepared					Failed					
	Reviewed By		Date Reviewed					Not Executable					
	Approved By		Date Approved					Defects Reported					
	TestCase Name		TestCase Objective					Tc'S Automated					
	Executed Name		Date Executed										
	Version		Build										
TC#	TS#	Test Design/Test Steps	Test Data	Expected Result	Actual Result	Pass	Fail	Not Executable	Severity				Defect ID
									Critical	Major	Modarate	Minar	

52. what is testing methodology?

Ans: Testing methodology is a systematic approach to software testing that outlines the tasks, techniques, and processes involved in testing. It serves as a blueprint for software testing and helps ensure that the testing process is systematic, efficient, and effective. The main objective of testing methodology is to find as many defects as possible in the software, within the given time and resources.

There are several different testing methodologies, each with its own strengths and weaknesses. Some of the most commonly used testing methodologies include:

Agile Testing: An iterative and incremental approach to testing that is commonly used in Agile software development.

Waterfall Testing: A sequential testing methodology that follows the Waterfall software development model.

V-Model Testing: A graphical representation of the testing process that maps each stage of the software development life cycle to its corresponding test activities.

Exploratory Testing: An approach to testing that focuses on discovering and exploring the software to uncover defects.

The choice of testing methodology depends on various factors, such as the size and complexity of the software, the goals of the testing process, and the available resources. The most important factor is that the methodology should be suitable for the software and should help the testing team achieve its objectives.

53. Difference between Smoke and Sanity?

Ans:

Smoke Testing	Sanity Testing
The main purpose of smoke testing is to quickly determine if the most critical functions of a software application are working correctly.	Sanity testing, on the other hand, is performed to verify if a small change or modification to the software has not affected its other functionalities.
Smoke testing is used to test all over function of the system/product.	Sanity testing is used in the case of only modified or defect functions of system/products.
This testing is performed by the developers or testers.	Sanity testing in software testing is usually performed by testers.
Smoke testing can be performed either manually or by using automation tools.	Sanity testing is commonly executed manually, not by using any automation approach.
Smoke Testing firstly performs on the initial build. smoke testing is done first.	Sanity Testing is done on stable builds or for the introduced new features in the software.

54. How to design Testcases and Scenarios?

Ans: How to create a Test Scenario:

⇒ Carefully study the Requirement Document – Business Requirement Specification (BRS), Software Requirement Specification (SRS), Functional Requirement Specification (FRS) pertaining to the System Under Test (SUT).

⇒ Isolate every requirement, and identify what possible user actions need to be tested for it. Figure out the technical issues associated with the requirement. Also, remember to analyze and frame possible system abuse scenarios by evaluating the software with a hacker's eyes.

⇒ Enumerate test scenarios that cover every possible feature of the software. Ensure that these scenarios cover every user flow and business flow involved in the operation of the website or app.

⇒ After listing the test scenarios, create a Traceability Matrix to ensure that every requirement is mapped to a test scenario.

⇒ Get the scenarios reviewed by a supervisor, and then push them to be reviewed by other stakeholders involved in the project.

How to create a Test Cases:

⇒ Simple and clear: Test cases need to be very concise, clear, and transparent. They should be easy and simple to understand not only for oneself but for others as well.

⇒ Maintain the uniqueness: While writing the test cases, it's necessary to make sure that they aren't being written over and over again and each case is different from the other.

⇒ Zero Assumptions: Test cases should not contain assumed data, don't come up with features/modules that don't exist.

⇒ Traceability: Test cases should be traceable for the future reference, so while writing it's important to keep that in mind,

⇒ Different input data: While writing test cases, all types of data must be taken into consideration.

⇒ Strong module name: The module name should be self-explanatory while writing the test case.

⇒ Minimal Description: The description of a test case should be small, one or two lines are normally considered good practice but it should give the basic overview properly.

⇒ Maximum conditions: All kinds of conditions should be taken into consideration while writing a test, increasing the effectiveness.

⇒ Meeting requirements: While writing the test case it's important that the client/customer/end-user requirements are met.

⇒ Repetitive Results: The test case must be written in such a way that it should provide the same result.

55. Test Case vs Test Scenario?

Ans: Test scenario is "What is to be tested". Identify all the possible areas to be tested.

Test case is “How, what is to be tested”.

Test Scenario	Test Case
A test scenario contains high-level documentation which describes an end-to-end functionality to be tested.	Test cases contain definite test steps, data, expected results for testing all the features of an application.
Test scenarios are derived from test artifacts like BRS, SRS, etc.	Test case is mostly derived from test scenarios. Multiple Test case can be derived from a single Test Scenario
It focuses on more “what to test” than “how to test”.	A complete emphasis on “what to test” and “how to test.”.
Comparatively less time and resources are required for creating & testing using scenarios.	More resources are needed for documentation and execution of test cases.
Test Scenario provides a small description, mostly one-line statements.	Test cases are more detailed with a number of parameters.

56. Difference between Defect, Error and Failure?

Ans:

Defect	**Error**	**Failure**
A Defect is a variance between expected and actual results. An Error that the tester finds is known as Defect.	The Error is a human mistake. An Error appears not only due to the logical mistake in the code made by the developer.	Failure is a consequence of a Defect. It is the observable incorrect behaviour of the system. Failure occurs when the software fails to perform in the real environment.

The Testers identify the defect. And it was also solved by the developer in the development phase or stage.	The Developers and automation test engineers raise the error.	The End-Users or Realtime users are identify Failures.

The Defect is the difference between the actual outcomes and expected outputs.	An Error is a mistake made in the code; that's why we cannot execute or compile code.	If the software has lots of defects, it leads to failure or causes failure.
The variation between the actual results and expected results is known as defect.	We can't compile or run a program due to coding mistake in a program. If a developer unable to successfully compile or run a program then they call it as an error.	Once the product is deployed and customers find any issues then they call the product as a failure product. After release, if an end user finds an issue then that particular issue is called as failure.

57. What is BVA and ECP?

Ans: **BVA:** Boundary Value Analysis is based on testing the boundary values of valid and invalid partitions. The behavior at the edge of the equivalence partition is more likely to be incorrect than the behavior within the partition, so boundaries are an area where testing is likely to yield defects.

Boundary value analysis is a testing technique used in software testing to identify errors in the boundary or edge values of input parameters. It is based on the assumption that most of the errors in a software system occur at the boundaries of the input domain.

The testing technique involves testing the system with input values that are on the boundary, just above the boundary, and just below the boundary. For example, if the input domain of a system is 1 to 100, the boundary values will be 1 and 100, and the values just above and below the boundary will be 2 and 99.

ECP: ECP (Equivalence Class Partitioning) is a software testing technique used to reduce the number of test cases required to test the software while maintaining the test coverage. It is based on the principle that a large number of test cases can be reduced to a smaller set of test cases by grouping the input data into equivalence classes. In ECP, the input domain is

dividedinto different equivalence classes, where each class represents a set of input values that are expected to behave in the same way by the software. The objective of ECP is to identify a representative value from each equivalence class and test it as a test case. This approach helps to reduce the number of test cases required to test the software while ensuring that each class is tested at least once.

For example, if the input domain of a software is a range of numbers from 1 to 100, it can be divided into several equivalence classes, such as valid numbers, invalid numbers, and boundary numbers. The valid numbers class can be further divided into sub-classes such as odd numbers and even numbers. The test cases can be selected from each of these equivalence classes and sub-classes to ensure that the software is tested for all possible input values.

58. What is functional testing and non-functional testing?

Ans: **Functional Testing** is a type of software testing that validates the software system against the functional requirements/specifications. The purpose of Functional tests is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements. Manual Testing or automation tools can be used for functional testing. The testing is done by providing sample inputs, capturing resulting outputs, and verifying that actual outputs are the same as expected outputs. Functional testing mainly involves black box testing and it is not concerned about the source code of the application. This testing checks User Interface, APIs, Database, Security, Client/Server communication and other functionality of the Application Under Test.

Non-Functional Testing is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc) of a software application. It is designed to test the readiness of a system as per non-functional parameters which are never addressed by functional testing. 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.

59. What are different types of SDLC models?

Ans: Software Development Life Cycle Models:

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models".

1. Waterfall Model
2. V Model:
3. Spiral Model
4. Agile Model

60. Name two parameters that can be useful to check the quality of test execution.

Ans: Two parameters required to check the quality of test execution include:

Defect leakage ratio: It represents the ratio of total potential rejections to the total overall production.

Defect reject ratio: It represents the ratio of total rejections to the total overall production.

61. When to choose manual testing over automation testing and vice versa?

Ans: Choosing Manual Testing over Automation Testing:

→ When test cases need to be run for a short duration of time (once or twice).
→ When one needs to perform exploratory testing, usability testing, or ad-hoc testing.
→ When assessing an application's user-friendliness.
→ Whenever flexibility is needed.
→ Whenever one wants to better manage complex situations/scenarios.

Choosing Automation Testing over Manual Testing

→ Whenever test cases have to be run repeatedly over a long period of time.
→ When one needs to perform performance testing, load testing, or regression testing.
→ Whenever one wishes to record the testing process.
→ When one has a limited amount of time to complete the testing phase.
→ When tests are needed to be executed in a standard runtime environment.

→ When tests involve repetitive steps.

→ When there are multiple and quick deployments for the product, the manual becomes very time taking and redundant.

62. In what way will you determine when to stop testing?

Ans: Testing can be quite challenging when it comes to determining when to stop. In the modern world, many software applications are so complex and run in so many interdependent environments, that complete testing is impossible. The following factors are often considered when deciding when to stop testing:

→ If deadlines are met (release deadlines, testing deadlines, etc.) and there are no high-priority issues left in the system.

→ Completion of test cases with a certain passing percentage.

→ As soon as the test budget is depleted.

→ The mean time between two inherent failures is known as the MTBF (Mean Time Between Failure). When the MTBF is quite high, the testing phase may be stopped depending on stakeholder decisions.

→ As soon as the automated code coverage meets a specified threshold value and there are no critical bugs.

→ If the bug rate drops below a certain level.

→ After the Beta or Alpha testing period has ended.

63. Can 100% testing coverage be achieved? How do you ensure test coverage?

Ans: Testing a product 100% is considered impossible. You can, however, get closer to your goal by following the steps below.

- Developing an effective testing strategy.
- Prepare a checklist for all activities related to testing.
- Establish a priority list for the application's critical areas.
- List all application requirements.
- Identify the risks associated with the application.
- Utilize automated testing

Agile Model Interview Questions and Answers

60. What is Agile Testing?

Ans: Agile Testing takes place in an environment where requirements keep changing according to the need of the customer. In this model, the testing and development activities are concurrent and the testing takes place throughout the development of the software. The testing team keeps receiving frequent small codes from the development team for testing.

61. Define the roles in Scrum?

Ans: There are mainly three roles that a Scrum team have:

1. Project Owner has the responsibility of managing the product backlog. Works with end-users and customers and provides proper requirements to the team to build the proper product.

2. Scrum Master works with the scrum team to make sure each sprint gets completed on time. Scrum master ensures proper workflow for the team.

3. Scrum Team: Each member of the team should be self-organized, dedicated and responsible for the high quality of the work.

62. What is Product Backlog & Sprint Backlog?

Ans: The Product backlog is maintained by the project owner which contains every feature and requirement of the product.

Sprint backlog can be treated as the subset of product backlog which contains features and requirements related to that particular sprint only.

63. Explain the difference between a traditional Waterfall model and Agile testing?

Ans: Agile testing is done parallel to the development activity whereas a traditional waterfall model testing is done at the end of the development.

As done in parallel, agile testing is done on small features whereas, in a waterfall model, testing is performed on the whole application.

64. Explain the Iterative and Incremental Development in Agile?

Ans: Iterative Development: Software is developed and delivered to the customer and based on the feedback again developed in cycles or releases and sprints. Example: Release 1 software is developed in 5 sprints and delivered to the customer. Now, the customer wants some changes, then the development team plan for 2nd release which can be completed in some sprints and so on.

Incremental Development: Software is developed in parts or increments. In each increment, a portion of the complete requirement is delivered.

65. What qualities should a good Agile tester have?

- He should be able to understand the requirements quickly.
- He should know Agile concepts and principals.
- As requirements keep changing, he should understand the risk involved in it.
- The agile tester should be able to prioritize the work based on the requirements.
- Communication is a must for an Agile tester as it requires a lot of communication with developers and business associates.

66. What is the difference between Epic, User stories & Tasks?

Ans:

1. User Stories: It defines the actual business requirement. Generally created by the business owner.
2. Task: To accomplish the business requirements development team create tasks.
3. Epic: A group of related user stories is called an Epic.

67. What is a Taskboard in Agile?

Ans: Taskboard is a dashboard that shows the progress of the project.

It contains:

User Story: It has the actual business requirement.

To Do: Tasks that can be worked on.

In Progress: Tasks in progress.

To Verify: Tasks pending for verification or testing

Done: Completed tasks.

68. What is Scrum ban?

Ans: It is a software development model that is a combination of Scrum and Kanban. Scrumban is considered for maintaining projects in which there are frequent changes or unexpected user stories. It can reduce the minimum completion time for user stories.

69. What is the Zero sprint in Agile?

Ans: It can be defined as a pre-preparation step to the first sprint. Activities like setting development environment, preparing backlog, etc need to be done before starting the first sprint and can be treated as Sprint zero.

70. What is the importance of daily stand up meetings?

Ans: Daily stand-up meeting is essential for any team in which team discuss,

How much work has been completed?

What are the plans to resolve technical issues?

What steps need to done to complete the projects etc?

71. How the velocity of the sprint is measured?

Ans: If capacity is measured as a percentage of a 40 hours weeks then, completed story points * team capacity

If capacity is measured in man-hours, then Completed story points/team capacity

72. How long the Scrum cycle last?

Ans: Basically, the Scrum cycle depends on the project size and team size. Team size may vary from 3 members to 9 members. Normally, it takes 3 to 4 weeks to complete a Scrum sprint. On an average, a scrum sprint ends in 4 weeks.

73. What is the scrum of scrums?

Ans: Suppose there are 7 teams working on a project and each team has 7 members. Each team leads its own particular scrum meeting. Now to coordinate among the teams a separate meeting has to be organized, that meeting is called Scrum of Scrums.

Scrum of Scrums:

An ambassador (a designated person who represents team) represents its team in the scrum of scrums.

Few points discussed in the meeting are:

The progress of the team, after the last meeting.

The task to be done before the next meeting.

Hindrance which the team had faced while completing the last task.

74. What are the principles of agile testing?

Ans: Some major principles of agile testing are:

- Customer satisfaction
- Bug-free clean code
- Changes are welcome by customer
- Whole team, business people and developers work collectively
- Instead of lengthy documentation, focus on the essence
- It focuses on face-to-face conversation
- It promotes sustainable development
- Deliver value to the customer
- Deliver working software frequently
- Simplified and clean code
- Practice continuous improvement
- Respond to change
- Focus on face-to-face conversation
- Less documentation
- Customer and developers work together collectively

- Promote sustainable development

75. What are the disadvantages of the agile model?

Ans: Some of the disadvantages of using the agile model are as follows:

- Not easy to predict: When you encounter a large project, it is not easy to get an idea of how much effort will it require.
- If the guidelines given by the customers are not properly grasped, then final outcome of the project is not as per customer satisfaction.
- Sometimes focusing on design and documentation is not proper
- High-level decisions are under the hand of Veterans, if not combined with non-experienced ones, freshers have little scope to grasp proper knowledge.

76. In what way does agile testing(development) methodology differ from the other testing(development) methodologies?

Ans: In Agile methodology, the code is broken into small parts and at a time, only that particular code is worked or tested. Continuous communication on the particular code part is done by a team so that the focus is only on that particular code. This makes the agile process more flexible and focused.

77. Explain what is a story point in the scrum?

Ans: It can be considered as a unit to estimate the total efforts required to complete or to do the particular task or implementing a backlog.

78. What are the main roles in the scrum?

Ans:

1. Scrum Team: Scrum team is made by an individual person who works collectively to achieve a particular task. The team works in a bond to deliver committed and requested products.
2. Scrum Master: Scrum Master is responsible for the proper execution or working of the scrum team. Being a servant – leader and a coach, he ensures the proper productivity of a team towards scrum sprint goal.
3. Product Owner: The product owner has the responsibility to deliver a complete picture

of what to build and to convey that idea to the team.

79. What is a product burndown chart?

Ans: A description in the form of the graph which shows implemented and not – implemented product backlog is called the burndown chart.

80. What is the defect burn down chart?

Ans: The number of defects identified and removed is represented by the defect burn down chart.

81. What is the sprint burndown chart?

Ans: A graph used to describe no. of implemented/non-implemented sprint in the Scrum cycle.

82. What is the Release burndown chart?

Ans: The graph used to depict the pending release which was earlier planned is called Release burn down the chart.

83. What is the sprint planning meeting?

Ans: A sprint planning meeting is joined by all entities like scrum master, product owner and whole scrum team where they discuss the priority features of the team and product backlog items.

84. What is a Sprint Retrospective meeting?

Ans: This is mostly the last part of the sprint or may be done after the sprint review meeting. Scrum master and the whole team participate in it. They discuss ‘ what was good during the sprint’,’ what was bad’,’ what needs to be improved’. It generally lasts for 2-3 hrs.

85. Tell me something about Kanban?

Ans: Kanban is a tool that helps the team to overlook the work i.e., its progress. Progress, as well as the status of your current development story, is perfectly described using Kanban and more accurately it is done by the ‘Kanban board’.

Kanban board allows you to write the whole scenario of your project at a single place so that you can get a perfect picture of the bottleneck, a task done, workflow progress or basically the complete status of your project.

86. Name three other Agile frameworks?

Ans: Test Driven Development, Feature Driven Development, and Kanban.

87. When to use agile model?

Ans: We can use agile model when the project is large, undefined, complex with unclear requirements. Agile methods are best suited for the project that requires frequent inspection by client so teams and stakeholders can assess and re-prioritize as needed to deliver the most value.

88. When not to use Agile?

Ans: The Project with a fixed scope and have everything pre-defined with little to no uncertainty don't require Agile methods.

Jira Tool Interview Questions and Answers

89. What is Jira?

Ans: JIRA is a software tool that was developed by the software company, Atlassian. JIRA is an issue and project tracking software to plan, track and manage your projects. JIRA is mainly used by agile development teams to customize your workflows, team collaboration, and release software with confidence. It is the perfect solution for organizing tasks and managing agile teams.

JIRA is an efficient tool and has the capability to track any kind of defects/issues.

90.Explain the workflow of JIRA?

Ans: JIRA Workflow is the series of stages or steps a bug/issue follows during its lifecycle from creation to closing or completion.

The important phases in the workflow are:

- Created/Open
- Work in Progress (WIP)
- Completed/Closed

91. What is an issue in JIRA?

Ans: An issue can be:

- A Software bug
- The Project Task
- The Form for Leave-request
- A Help-desk Ticket

92. What are some of the benefits of using Jira?

Ans: The following are some of the benefits of using Jira:

- It is easily customizable and extensible.
- It runs almost anywhere as it is platform-independent. It is recognized by quite a few well-known companies.
- We can get the latest update on the progress of projects via Jira.
- It has an upfront and fair licensing policy.

93. What are the agile methodologies supported by Jira?

Ans: Jira Software supports the following agile methodologies: Scrum and Kanban.

Scrum: Scrum is an agile methodology in which the development team works in an iterative manner to complete a given project. Every sprint or iteration has some set scope and timeline for the project. For software development projects, Scrum is most suitable.

Kanban: Kanban is an agile methodology that focuses on just-in-time delivery. It accomplishes this by visualizing the workflow and the tasks in progress. Kanban is mostly suitable for operation teams.

94. What are the issue types in a Scrum project in Jira?

Ans: An individual unit of work in Jira is referred to as an issue. This issue could be a unit such

as a story or an epic. Every issue has a field called issue type, which represents the type and use of the issue. For instance, in a Scrum project, one would have the following types of issues by default:

Story: A story in Jira represents a single feature that is to be implemented. It is generally used in order to get the requirements from the end user's perspective. For this purpose, stories are often written in non-technical language. These are used for focusing on the end results of the feature.

Epic: An epic means a big user story. This represents that the story has not been broken down into smaller and finer requirements. In Jira, epics are mostly used to define the "theme" for various stories that will be part of it, along with the modules or the major components in a big development project.

Bug: A bug in Jira represents a problem or a defect that needs to be fixed in a given product.

Task: This represents a generic task that is neither a bug nor a story, but it needs to be completed.

95. What are some of the popular add-ons for Jira?

Ans: Following are some of the popular add-ons for JIRA:

- Zephyr for JIRA – Test Management
- JIRA Toolkit Plugin
- JIRA Charting Plugin
- Portfolio for JIRA
- Suite's utilities for JIRA
- ScriptRunner for JIRA
- Atlassian REST API Browser
- Tempo Timesheets for JIRA
- JIRA Misc Workflow Extensions

96. What is an Event in Jira?

Ans: In Jira, an event gives information about the status, the default template and the

notification scheme and workflow transition post function associations for the event. Basically, the events are classified into these two categories:

- A System event (Jira defined events)
- Custom event (User-defined events)

97. What are the report types that are generated in Jira?

Ans: In Jira, there are a number of reports generated that are used to showcase different project statistics throughout the entire life cycle of the project. There are general reports available for analyzing issues in Jira. Along with this, there are also some reports for Scrum and Kanban projects.

The general reports available for analyzing the issues includes:	**For Scrum projects, the following types of reports can be generated:**
Time Tracking Report	Sprint Report
Version Workload Report	Version Report
Workload Pie chart Report	Control chart
Created vs Resolved issue Report	Velocity chart
Average Age Report	Cumulative Flow diagram
User Workload Report	Release Burndown
Resolution Time Report	Burndown chart
Pie Chart Report	Epic Report
Recently created Issue Report	

98. Explain the three colour indicators to show time spent in Jira and their significance.

Ans: In Jira, for a particular issue, three colour, s namely Blue, Green, and Orange are used to show how much time is spent on that particular issue. You can go to the section of 'Time Tracking' to view this information. The significance of each colour is as follows:

Blue: The blue colour is used to show the 'Original Estimate'. This is the estimate of time to be spent in resolving an issue. You can see this field shown as 'Estimated'.

Orange: The orange colour is for the time left to resolve an issue. You can see this field shown

as ‘Remaining’.

Green: The green colour shows the actual time that has been spent in solving given the issue till now. You can see this field shown as ‘logged’.

99. For an Agile project, how are user stories created in Jira?

Ans: For an Agile project, the user stories are created in the following way in Jira:

Issue type -> Epic and Issue type –> Story linked to it. To do so, on the ‘Create Issue’ page, go to “Configure Fields”. Then select the “Epic Link” field to be included in the issue creation screen.

Alternatively, you can also have a product backlog by creating one main User story and having multiple sub-tasks under it.

100. Why are the issues in Jira labelled?

Ans: The issues in Jira are labelled in order to categorize an issue within a particular section. This can help to easily search with the help of labels. Label for a given issue can be initially set at the time of creating the issue. It can also be edited within the issue.

101. What does the Jira Schema consist of?

Ans: The Jira Schema consists of the following:

Notifications: This indicates what email someone receives when an issue is changed.

Workflows: This indicates which workflow is used for each issue type.

Permissions: This indicates what changes can be made to an issue by someone.

Issue types: This indicates what issue types like Bug, New Feature etc. can be used in a given Jira project.

Screens: This indicates where the fields are displayed in an issue’s screen.

Field configurations: This is used to define which Field Configuration is used for each issue type.

102. What are some of the common Jira add-ons?

Ans: Some of the common and useful Jira add-ons are as follows:

1. Slack
2. Github
3. Tempo Timesheets
4. PagerDuty
5. Jenkins-CI
6. Usersnap

103. What do you mean by Cloning an Issue in JIRA?

Ans: Cloning an issue basically means copying an issue. This allows us to quickly make a copy of an issue inside a similar project. The clone issue is a mirror image of the original issue. It contains similar data that is put in the original issue like the Summary, Components, Affects Versions, and so on. A clone issue is a different element from the original issue. The clone issue can similarly be connected to the original issue.

The tasks on the original issue have no impact on the clone issue and also the other way around. If a link is made between the original issue and the clone issue, that serves as the main connection.

Cloning an issue retains the following:

- Project
- Summary
- Issue Links
- Description
- Assignee
- Environment
- Reporter
- Attachments
- Components
- Affects Versions
- Fix For Versions
- Priority

- Issue Type
- Security

104. Which issues cannot be cloned?

Ans: The following are the issues that are cannot be cloned:

- Time tracking
- Issue history
- Comments and
- Links to Confluence pages

105.Can we back up data in Jira Cloud?

Ans: Yes, Jira provides backup functionality of data. But it can save backup files only once and next time when you backup, the existing data will be overwritten.

106. What is scheduling an event in Jira?

Ans: Scheduling an event is activated in order to trigger action with respect to the issue. And to perform this scheduling, one should request "Schedule issue permission" from the administrator. This provides a "due date" to an issue to be scheduled for.

107. What are the important differences between JIRA and Bugzilla?

Ans: Let me explain a few important differences between JIRA and Bugzilla

JIRA	**Bugzilla**
JIRA is a commercial software tool	Bugzilla is an open-source software tool
JIRA is a user-friendly software tool	Bugzilla is not a user-friendly tool
Drag and drop issue options are available in JIRA	Drag and drop issue options are not available in Bugzilla
JIRA has a configuration link types which contains user-defined semantics	Bugzilla has only one configuration link type