1. What is Exploratory Testing :

Exploratory testing is a software testing approach where testers actively explore the application without predefined test cases, using their creativity, experience, and critical thinking to discover issues.

- Test design, execution and logging happen simultaneously
- Testing is often not recorded
- Makes use of experience and test patterns
- The focus of exploratory testing is more on testing as a “thinking” activity.

2. What is traceability matrix :

A traceability matrix is a document that maps test cases to requirements .

- To make sure that all requirements included in the test cases
- To make sure that developers are not creating features that no one has requested
- Easy to identify the missing functionalities.
- If there is a change request for a requirement, then we can easily find out which test cases need to update

3. What is Boundary value testing :

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

- At those points when input values change from valid to invalid errors

EX : [-1 { invalid value} 0 1 {valid value}] to [99 {valid value} 100 {invalid value}] . Range 0 to 100

4. What is Equivalence partitioning testing :

The numbers fall into a partition where each would have the same, or equivalent, result i.e. an Equivalence Partition .

EP says that by testing just one value we have tested the partition (typically a mid-point value is used :

- If one value finds a bug, the others probably will too
- If one doesn't find a bug, the others probably won't either

EX : { 1 15 21 22 35 57 68 89 100 } i.e; 35 is tested .

5. What is Integration testing :

Integration Testing is a level of the software testing process
where individual units are combined and tested as a group.

The goal is to ensure that these components work together as expected when integrated into a whole system

6 . What determines the level of risk

- Testing is necessary because we all make mistakes.
- Some of those mistakes are unimportant, but some of them are expensive or dangerous .
- However some mistakes come from bad assumptions and blind spots.
- We need to check everything and anything we produce because things can always go wrong – humans make mistakes all the time.

7. What is Alpha testing :

Alpha testing is **the initial phase of validating whether a new product will perform as expected .**

- It is always performed by the developers and testers at the software development site.
- Alpha testing helps identify and fix bugs before the product is released to the public .

8. What is beta testing :

Beta testing is **the process of testing a software product or service in a real-world environment** to uncover any bugs or issues before a official release .

- Beta Testing is always open to the market and public.
- It is also considered as the User Acceptance Testing (UAT) which is done at customers.
- Beta testing can be considered “pre-release” testing.

9. What is component testing :

A minimal software item that can be tested in isolation.

- It means “A unit is the smallest testable part of software.”
- Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.

- Unit testing is performed by using the White Box Testing method.

10. What is functional system testing :

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

- A Requirement may exist as a text document
- A requirement that specifies a function that a system or system component must perform .

11. What is Non-Functional Testing :

Non-Functional testing check the performance , reliability, scalability and other non-functional aspects of the system .

- It is the testing of “how” the system works .
- Non-functional testing may be performed at all test levels .

12. What is Graphical User Interface (GUI) Testing :

GUI Testing (Graphical User Interface Testing) is a type of software testing that focuses on verifying the **graphical user interface** of an application to ensure it functions correctly and provides a good user experience .

- GUI testing is to validate the visual elements of an application, such as buttons, icons, menus, forms, and other interactive components, ensuring that they are intuitive, responsive, and behave as expected.

13.What is Adhoc testing :

Adhoc testing is an informal testing type with an aim to break the system .

- Main aim of this testing is to find defects by random checking.
- Adhoc testing can be achieved with the testing technique called Error Guessing.
- In fact it does not create test cases altogether!
- Error guessing can be done by the people having enough experience on the system to “guess” the most likely source of errors.

14. What is load testing :

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.

- Load testing is commonly used for the Client/Server, Web based applications .
- Determine whether current infrastructure is sufficient to run the application .
- Sustainability of application with respect to peak user load .

15. What is stress Testing :

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.

16 . 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’** or **‘open box’** testing technique .
- Here the testers require knowledge of how the software is implemented, “how it works” .
- Testing based upon the structure of the code
- The tester needs to have a look inside the source code .

Types Of White Box testing :
1. Statement coverage
2. Decision coverage
3. Condition coverage

1. <u>Statement coverage</u> :

- In this process each and every line of code needs to be checked and executed.
- The statement coverage covers only the true conditions.
- Through statement coverage we can identify the statements executed and where the code is not executed because of blockage

Advantage :
- It verifies what the written code is expected to do and not to do .
- It measures the quality of code written .
- It checks the flow of different paths in the program and it also ensure that whether those paths are tested or not.

Disadvantage :
- It cannot test the false conditions.
- It does not report whether the loop reaches its termination condition .
- It does not understand the logical operators.

.

2. <u>Decision coverage</u> :

- Decision coverage also known as branch coverage or all-edges coverage.
- It covers both the true and false conditions unlikely the statement coverage.
- Aim is to demonstrate that all Decisions have been run at least once .
- With an IF statement, the exit can either be TRUE or FALSE, depending on the value of the logical condition that comes after IF.

<u>Advantage :</u>

- To validate that all the branches in the code are reached .
- It eliminates problems that occur with statement coverage testing .

<u>Disadvantage :</u>

- This metric ignores branches within Boolean expressions which occur due to short-circuit operators .

3. <u>Condition coverage</u> :

- This is closely related to decision coverage but has better sensitivity to the control flow.
- However, full condition coverage does not guarantee full decision coverage.
- Condition coverage reports the true or false outcome of each condition.
- Condition coverage measures the conditions independently of each other.

17. 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.

- The testers have no knowledge of how the system or component is structured inside the box .
- The technique of testing without having any knowledge of the interior workings of the application is Black Box testing.
- What a system does, rather than HOW it does it .

Advantages :

- Well suited and efficient for large code segments.
- Code Access not required.

Disadvantages :

- Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
- Blind Coverage, since the tester cannot target specific code segments or error prone areas.
- The test cases are difficult to design.

Techniques of Black Box Testing :

1. Equivalence partitioning
2. Boundary value analysis
3. Decision tables
4. State transition testing

1.Equivalence partitioning :

- EP says that by testing just one value we have tested the partition (typically a mid-point value is used). It assumes that:
  > If one value finds a bug, the others probably will too .
  > If one doesn't find a bug, the others probably won't either .

- The Valid partition is bounded by the values 1 and 100 .
- Plus there are 2 Invalid partitions .

2. Boundary value analysis :

- 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
- At those points when input values change from valid to invalid errors are most likely to occur .

3. Decision Table :

- Decision Table the relationships between the inputs and the possible outputs are mapped together .
  Table based technique where :
- Inputs to the system are recorded
- Outputs to the system are defined

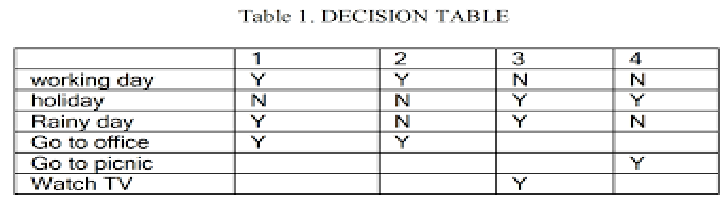
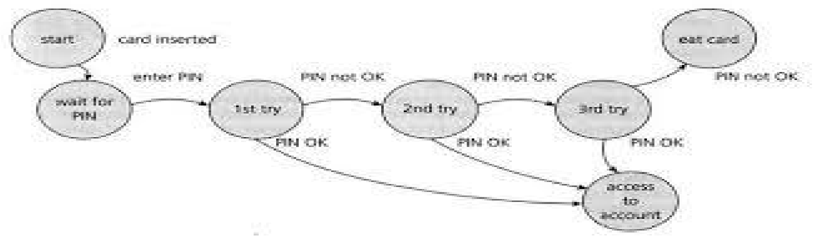


Table 1. DECISION TABLE

	1	2	3	4
working day	Y	Y	N	N
holiday	N	N	Y	Y
Rainy day	Y	N	Y	N
Go to office	Y	Y		
Go to picnic				Y
Watch TV			Y	

4.State transition testing :

- **State Transition**: A transition between two states of a component or system .
- **State Transition Testing:** A black box test design technique in which test cases are designed to execute valid and invalid state transitions. Also known as N-switch testing.
- Any system where you get a different output for the same input, depending on what has happened before, is a finite state system.



18. 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 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:
  ⚫ All JavaScript errors
  ⚫ Buttons like Save, Delete, Cancel not performing their intended functions
  ⚫ A missing functionality (or) a feature not functioning the way it is intended to

- **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

19. Mention what Bigbang testing :

Big bang integration testing is a testing approach where all components or modules are integrated and tested as a single unit .

- All components or modules are integrated simultaneously, after which everything is tested as a whole .

Advantages:

- Everything is finished before integration testing starts
- Convenient for small systems .

Disadvantages :

- In general it is time consuming and difficult to trace the cause of failures because of this late integration.
- Since all modules are tested at once, high risk critical modules are not isolated and tested on priority.

20.What is the purpose of exit criteria :

"When to stop testing" is one of the most difficult questions to a test engineer. The following are few of the common Test Stop criteria:

- ⚫ All the high priority bugs are fixed.
- ⚫ The rate at which bugs are found is too small.
- ⚫ The testing budget is exhausted.
- ⚫ The project duration is completed.
- ⚫ The risk in the project is under acceptable limit.
- ⚫ Run out of time
- ⚫ Run out of budget

⚫ Boss says stop
⚫ All defects have been fixed

21.When should "Regression Testing" be performed :

Regression testing : Regression testing means testing your software application when it undergoes a code change to ensure that the new code has not affected other parts of the software .

- This testing is done to make sure that new code changes should not have side effects on the existing functionalities.

Regression testing should be carried out :

- Change in requirements and code is modified according to the requirement
- New feature is added to the software
- Defect fixing
- Performance issue fix

22.What is Error, Defect, Bug and failure :

- A mistake in coding is called an error .
- Error found by a tester is called a defect .
- Defect accepted by development team then it is called a bug .

- Build does not meet the requirements then it is failure .

23. What are 7 key principles? Explain in detail :

7 key principles :

1. Testing shows presence of Defects
2. Exhaustive Testing is Impossible!
3. Early Testing
4. Defect Clustering
5. The Pesticide Paradox
6. Testing is Context Dependent
7. Absence of Errors Fallacy

1. Testing shows presence of Defects :

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

2. Exhaustive Testing is Impossible!

- It's impossible to test all possible combinations of inputs & outputs .

- Is too expensive , it takes too long to provide results .
- We must prioritise our testing effort using a **Risk Based Approach** .

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

3. Early Testing :

- Testing activities should start as early as possible in the software or system development life cycle .
- Early testing helps identify defects early , reducing the cost of fixing them later .
- Testing doesn’t start once the code has been written .

4. Defect Clustering :

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

5. The Pesticide Paradox :

- Running 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 PP , test cases need to be regularly reviewed , updated & new test cases added to find new defect’s .
- Testing identifies bugs, and programmers respond to fix them .
-

6. Testing is Context Dependent :

- Testing is basically context dependent.
- Testing is done differently in different contexts
- Different kinds of sites are tested differently.

7. Absence of Errors Fallacy :

- Even if a software has no error's , it may still not meet users' needs or expectations .
- Testing should focus on ensuring the software meets its functional & non functional requirements .
- If we build a system, doing well , find & fix defects.

24. Difference between QA v/s QC v/s Tester :

< < <

Quality Assurance VS	**Quality Control** VS	**Testing**
Preventing defects & ensuring that process & standards are followed throughout the development lifecycle	Identifying & correcting defects in the software product .	Executing test cases to verify the functionality & performance of the software .
Prevention	Detection	Verification
Defining standards , reviewing processes & conducting audits process improvement .	Testing , code reviews & inspections	Designing , executing & analyzing test cases .
Proactive	Reactive	Execution oriented
It ensure entire Software development life cycle	It ensures specific phases of development .	It ensure specific testing activities .
Process oriented activities	Product oriented activities	Product oriented activities

25. What is Bug Life Cycle :

"A computer bug is an error, flaw, mistake, failure, or fault in a

computer program that prevents it from working correctly or produces an incorrect result. Bugs arise from mistakes and errors, made by people, in either a program's source code or its design. "
The duration or time span between the first time defects is found and the time that it is closed successfully, rejected, postponed or deferred is called as 'BUG (defect) Life Cycle'

26.Difference between Smoke and Sanity :

Smoke Testing vs	**Sanity Testing**
Smoke Testing is performed to ascertain that the critical functionalities of the program are working fine .	**Sanity Testing is done to check the new functionality / bugs have been fixed .**
The objective of this testing is to verify "stability" of the system in order to withstand more rigorous testing .	**The objective of the testing is to verify the "rationality" of the system in order to proceed with more rigorous testing .**
This testing is performed by the developers or testers .	**Sanity testing is usually performed by testers**
Smoke testing is usually documented or scripted .	**Sanity testing is usually not documented and is unscripted .**
Smoke testing is a subset of Acceptance testing .	**Sanity testing is a subset of Regression testing .**
Smoke testing exercises the entire system from end to end .	**Sanity testing exercises only the particular component of the entire system .**
Smoke testing is like General Health Check up	**Sanity Testing is like specialized health checkup .**

27 .Difference between Priority and Severity

While **severity** refers to the impact of a defect (how serious or harmful it is to the software's functionality), **priority** refers to how soon the defect needs to be fixed (how urgent the resolution is). For example, a defect might have high severity (e.g., a system crash) but low priority (if it only occurs in rare, non-critical circumstances).

28. Difference between verification and Validation :

Verification	Validation
Def : The process of evaluating work-products (not the actual final product) of a development phase to determine whether they meet the specified requirements for that phase .	**Def** : The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements.
Obj : To ensure that the product is being built according to the requirements and design specifications. In other words, to ensure that work products meet their specified requirements .	**Obj** : To ensure that the product actually meets the user's needs, and that the specifications were correct in the first place. In other words, to demonstrate that the product fulfills its intended use when placed in its intended environment.
Are we building the product right?	Are we building the right product?
Plans, Requirement Specs, Design Specs, Code, Test Cases	The actual product/software.
Reviews · Walkthroughs · Inspections	Testing

29 . Explain types of Performance testing :

Performance testing : Software performance testing is a means of quality assurance (QA). It involves testing software applications to ensure they will perform well under their expected workload .

The focus of Performance testing is checking a software programs :

- **Speed** – Determines whether the application responds quickly
- **Scalability** – Determines maximum user load the software application can handle it.
- **Stability** – Determines if the application is stable under varying loads

Types of Performance Testing :

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

1. **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 .

- This testing helps determine how the application behaves when multiple users access it simultaneously .

2. **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.
- Stress testing is also known as **Endurance testing .**

3. Endurance testing : Endurance testing is a type of performance test that assesses how a product or system performs under prolonged stress.

- Type of non-functional testing where your system is subjected to a significant load over an extended period.

4 . Spike testing :

- Type of performance testing in which an application receives a sudden and extreme increase or decrease in load.
- The goal of spike testing is to determine the behavior of a software application when it receives extreme variations in traffic.

5 . Volume testing : A software testing method that evaluates a system's stability and response time by loading it with large amounts of data .

- Examines the stability and response time of a system by transferring huge volumes of data .

6. Scalability testing : Scalability testing is a type of performance testing that evaluates how well a system can handle increased loads .

- **Scale testing** is used to identify the performance of a system or network when the number of users, devices, data points or transactions on a product increases **.**

30. To create HLR & TestCase of (Instagram & Facebook) for first page and chat functionality .

Test case Facebook
Test case Instagram

31 . Explain the difference between Functional testing and NonFunctional testing :

Functional testing	VS Non Functional testing
Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements.	Non-Functional testing checks the Performance, reliability, scalability and other non-functional aspects of the software system.
Functional testing is executed first	Non functional testing should be performed after functional testing
Manual testing or automation tools can be used for functional testing	Using tools will be effective for this testing
Business requirements are the inputs to functional testing	Performance parameters like speed , scalability are inputs to non-functional testing.
Functional testing describes what the product does	Nonfunctional testing describes how good the product works
Easy to do manual testing	Tough to do manual testing
Types of Functional testing are	Types of Nonfunctional testing are

· Unit Testing · Smoke Testing · Sanity Testing · Integration Testing · White box testing · Black Box testing · User Acceptance testing · Regression Testing	· Performance Testing · Load Testing · Volume Testing · Stress Testing · Security Testing · Installation Testing · Penetration Testing · Compatibility Testing · Migration Testing

32. What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle) :

SDLC	STLC
Requirements Collection/Gathering	Requirement Analysis
Analysis	Test Planning
Design	Test case development
Implementation	Test Environment setup
Testing	Test Execution
Maintenance	Test Cycle closure

33. What is the difference between test scenarios, test cases, and test script :

Test Scenario :

A Scenario is any functionality that can be tested. It is also called Test Condition, or Test Possibility.

- Test Scenario is ‘What to be tested’
- Test scenario is nothing but a test procedure.
- The scenarios are derived from use cases.
- Test Scenario represents a series of actions that are associated together.
- Scenario is thread of operations .

Test Case :

Test cases involve the set of steps, conditions and inputs which can be used while performing the testing tasks.

- Test Case is ‘How to be tested’
- Test cases are derived (or written) from test scenarios.
- Test Case represents a single (low level) action by the user.
- Test cases are a set of input and output given to the System.

Formal Test Template :

- ☐ **Test case id**
- ☐ **Product module id**
- ☐ **Test case description**
- ☐ **Test steps**
- ☐ **Expected result**
- ☐ **Actual result**
- ☐ **Post condition’s (Pass / Fail)**

Test Data : The Input data describes what needs to be entered for a step .

Test Script :

A test script in software testing is a set of instructions that will be performed on the system under test to test that the system functions as expected.

> Manual Testing

> Automation Testing

- One script is written to explain how to simulate each business scenario .
- Written to a level of detail for which someone else would be able to easily execute .
- Identified the input/test data that should be entered for each transaction .
- Identifies the test condition that is being satisfied for each step, if applicable .

# **Test script** is a set of instructions that manual & automates the testing of a software application's functionality .

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

Test Plan : The overall approach of testing (the test strategy), including the definition of the test levels and entry and exit criteria.

Test Planning in **STLC** is a phase in which a **Senior QA manager** determines the test plan strategy along with efforts and cost estimates for the project .

Activities :

- Preparation of test plan/strategy document for various types of testing
- Test tool selection
- Test effort estimation
- Resource planning and determining roles and responsibilities.
- Training requirement

35 . What is priority :

Priority : Priority defines the order in which we should resolve a defect. Should we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mentioned then the developer has to fix it at the earliest. The priority status is set based on the customer requirements .

For example: If the company name is misspelled in the home page of the website, then the priority is high and severity is low to fix it .

36 . What is Severity :

Severity : It is the extent to which the defect can affect the software. In other words it defines the impact that a given defect has on the system .
For example: If an application or web page crashes when a remote link is clicked, in this case clicking the remote link by a user is rare but the impact of application crashing is severe. So the severity is high but
priority is low

37. Bug and categories :

A **bug** in software development refers to a flaw, fault, or unintended behavior in a software application or system that causes it to function incorrectly or unexpectedly.

Types of Bugs:

1. **Functional Bugs**: These affect how the software behaves or operates. The software doesn't perform the expected task or produces incorrect results.
2. **UI/UX Bugs**: Related to the user interface and user experience, such as incorrect layout, color schemes, or usability issues.
3. **Performance Bugs**: These result in slow performance, high resource consumption, or crashes.
4. **Security Bugs**: Vulnerabilities that expose the software to potential security threats.
5. **Compatibility Bugs**: Issues that occur when the software doesn't work across different devices, browsers, or operating systems.
6. **Regression Bugs**: Bugs that appear after changes to the software, affecting previously working functionality.
7. **Integration Bugs**: Occur when different parts of the system or external systems fail to work together as intended.
8. **Concurrency Bugs**: Issues that arise in multi-threaded or multi-user environments.

Bug Life Cycle:

1. **Identification**: A bug is discovered through testing, user reports, or other means.
2. **Reporting**: The bug is documented in a bug-tracking system with relevant details (steps to reproduce, environment, etc.).
3. **Prioritization**: The bug is categorized based on its severity and priority (how urgently it needs to be fixed).
4. **Fixing**: The development team works to correct the issue.
5. **Verification**: After the fix, the bug is tested to ensure it's resolved and no new issues are introduced.
6. **Closure**: Once verified, the bug is marked as resolved and closed in the tracking system.

Bug Severity vs. Priority:

- **Severity**: How serious the bug is (e.g., critical, major, minor).
- **Priority**: How quickly the bug needs to be addressed, which may differ from its severity (e.g., high, medium, low).

38 . Advantage of Bugzilla :

Bugzilla is an open-source, web-based bug tracking system used by software development teams to track and manage bugs, issues, and tasks in a software project. It helps organize the process of identifying, reporting, prioritizing, and resolving defects in the software development lifecycle. Bugzilla is highly customizable, scalable, and supports teams in various industries, from small startups to large enterprises.

Advantages of Bugzilla:

- **Customizable**: Tailors to the specific needs of different teams or organizations.
- **Open Source**: Free to use, and the source code is available for modification.
- **Scalability**: Suitable for small and large teams, handling a high volume of bugs and users.
- **Web-Based**: Accessible from anywhere with internet access.
- **Extensive Documentation**: Comprehensive user manuals and online resources are available to support users.

Common Workflow in Bugzilla:

1. **Bug Reporting**:
   - A user or tester reports a bug by filling out a form with details like bug description, environment, severity, priority, steps to reproduce, and expected vs. actual behavior.
2. **Bug Assignment**:
   - Once the bug is reported, it is assigned to a developer or team member to investigate and fix. The person assigned can update the bug status.

3. **Bug Fixing**:
   - The developer works on fixing the bug, which may involve writing code, performing tests, and validating the issue.
4. **Bug Verification**:
   - After the fix, the tester verifies the resolution by retesting the bug in the software. If the issue is resolved, the bug is closed.
5. **Bug Closure**:
   - Once the fix is verified, the bug is marked as closed or resolved in Bugzilla.

39.What are the different Methodologies in Agile Development Model :

Agile model is a combination of iterative & incremental process models with focus on process adaptability & customer satisfaction by rapid delivery of working software products .

- It breaks the product into small incremental build’s & build’s are provided in iterations .

- Every iteration involves cross functional teams working simultaneously on various areas like planning, requirement analysis, design, coding, unit testing & acceptance testing .

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

Authorization: Authorization is the process of determining what an authenticated user is allowed to do. It defines the access level or permissions granted to the user based on their identity or role.

Authentication: Authentication is the process of verifying the identity of a user, device, or system. It ensures that the entity requesting access is who they claim to be.

Differences Between Authentication and Authorization:

Aspect	**Authentication**	**Authorization**
Purpose	Verifies **who** the user is.	Determines **what** the user can do.
When It Occurs	Happens **first**, before authorization.	Happens **after** authentication.
Data Used	Uses credentials (e.g., username, password, biometrics).	Uses permissions or roles assigned to the user.
Outcome	Validates the identity of the user.	Grants or denies access to resources or actions.
Example	Logging in with your username and password.	Being allowed to edit a document based on your role.

41. To create HLR & TestCase of WebBased (WhatsApp web) :

1. WhatsApp Web : https://web.whatsapp.com

https://docs.google.com/spreadsheets/d/1dWTdiwspw4EatSg_BnA3BEyIsBechz0WjO_PISw1SFM/edit?usp=sharing

42 .Create TestCases on Whatsapp Group Chat :

https://docs.google.com/spreadsheets/d/1m__216uuEPIAhBLJaut195s8WIEgxbsHoci9s2HRghQ/edit?usp=sharing

43 . Write a scenario of only Whatsapp chat messages :

1. Verify that the user can set a chat wallpaper.
2. Verify that the user sets privacy settings like turning on/off last seen, online status, read receipts, etc.
3. Verify that the user can update notification settings like – notification sound, on/off, and show preview for both group and individual chats.

4. Verify that the user can take the complete chat backup of his chats.
5. Verify that the user can update the phone number that is used by the WhatsApp application.
6. Verify that the user can disable/delete his Whatsapp account.
7. Verify that the user can check data usage by images, audio, video, and documents in WhatsApp chats.

44. Write a Scenario of Pen :

1. Verify the type of pen, whether it is a ballpoint pen, ink pen, or gel pen.
2. Verify that the user is able to write clearly over different types of papers.
3. Check the weight of the pen. It should be as per the specifications. In case not mentioned in the specifications, the weight should not be too heavy to impact its smooth operation.
4. Verify if the pen is with a cap or without a cap.
5. Verify the color of the ink on the pen.
6. Check the odor of the pen’s ink on writing over a surface.
7. Verify the surfaces over which the pen is able to write smoothly apart from paper e.g. cardboard, rubber surface, etc.
8. Verify that the text written by the pen should have consistent ink flow without leaving any blob.
9. Check that the pen’s ink should not leak in case it is tilted upside down.
10. Verify if the pen’s ink should not leak at higher altitudes.
11. Verify if the text written by the pen is erasable or not.
12. Check the functioning of the pen by applying normal pressure during writing.
13. Verify the strength of the pen’s outer body. It should not be easily breakable.
14. Verify that text written by pen should not get faded before a certain time as mentioned in the specification.
15. Check if the text written by the pen is waterproof or not.

16. Verify that the user is able to write normally by tilting the pen at a certain angle instead of keeping it straight while writing.
17. Check the grip of the pen, and whether it provides adequate friction for the user to comfortably grip the pen.
18. Verify if the pen can support multiple refills or not.
19. In the case of an ink pen, verify that the user is able to refill the pen with all the supported ink types.
20. For ink pens, verify that the mechanism to refill the pen is easy to operate.
21. In the case of a ballpoint pen, verify the size of the tip.
22. In the case of a ball and gel pen, verify that the user can change the refill of the pen easily.

45. Write a Scenario of Door :

1. Verify if the door is a single door or bi-folded door.
2. Check if the door opens inwards or outwards.
3. Verify that the dimension of the doors are as per the specifications.
4. Verify that the material used in the door body and its parts is as per the specifications.
5. Verify that the color of the door is as specified.
6. Verify if the door is a sliding door or rotating door.
7. Check the position, quality and strength of hinges.
8. Check the type of locks in the door.
9. Check the number of locks in the door interior side or exterior side.
10. Verify if the door is having a peek-hole or not.
11. Verify if the door has a stopper or not.
12. Verify if the door closes automatically or not – spring mechanism.
13. Verify if the door makes noise when opened or closed.
14. Check the door condition when used extensively with water.
15. Check the door condition in different climatic conditions- temperature, humidity etc.
16. Check the amount of force- pull or push required to open or close the door.

46. Write a Scenario of ATM :

1. Verify the type of ATM machine, if it has a touch screen, both keypad buttons only, or both.
2. Verify that on properly inserting a valid card different banking options appear on the screen.
3. Check that no option to continue and enter credentials is displayed to the user when the card is inserted incorrectly.
4. Verify that the touch of the ATM screen is smooth and operational.
5. Verify that the user is presented with the option to choose a language for further operations.
6. Check that the user is asked to enter a pin number before displaying any card/bank account detail.

7. Verify that there is a limited number of attempts up to which the user is allowed to enter the pin code.
8. Verify that if the total number of incorrect pin attempts gets surpassed then the user is not allowed to continue further. And operations like temporary blocking of the card, etc get initiated.
9. Check that the pin is displayed in masked form when entered.
10. Verify that the user is presented with different account type options like- saving, current, etc.
11. Verify that the user is allowed to get account details like available balance.
12. Check that the correct amount of money gets withdrawn as entered by the user for cash withdrawal.
13. Verify that the user is only allowed to enter the amount in multiple denominations as per the specifications.
14. Verify that the user is prompted to enter the amount again in case the amount entered is less than the minimum amount configured.
15. Check that the user cannot withdraw more amount than the total available balance and a proper message should be displayed.
16. Verify that the user is provided the option to get the transaction details in printed form.
17. Verify that the user’s session timeout is maintained.
18. Check that the user is not allowed to exceed one transaction limit amount.

19. Verify that the user is not allowed to exceed the one-day transaction limit amount.
20. Verify that the user is allowed to do only one transaction per pin request.
21. Check that in case the ATM machine runs out of money, a proper message is displayed to the user.
22. Verify that the applicable fee gets deducted along with the withdrawn amount in case the user exceeds the limit of the number of free transactions in a month.
23. Verify that the applicable fee gets deducted along with the withdrawn amount in case the user uses a card of a bank other than that of an ATM.
24. Check that the user is not allowed to proceed with the expired ATM card and that a proper error message gets displayed.
25. Verify that in case of sudden electricity loss before withdrawing cash, the transaction is marked as null and the amount is not withdrawn from the user's account.

47. When to used Usability Testing :

There are many software applications / websites, which miserably fail,once launched, due to following reasons –

- Where do I click next?
- Which page needs to be navigated?
- Which Icon or Jargon represents what?
- Error messages are not consistent or effectively displayed
- Session time is not sufficient.
- Usability Testing identifies usability errors in the system early in development cycle and can save a product from failure .

Usability Testing Example :

Web Based Testing , Desktop Based , Mobile based & Game based Testing

- All fields on a page (For Example, text box, radio options, drop-down lists) should be aligned properly.
- The user should not be able to type in drop-down select lists.
- Tab and Shift+Tab order should work properly.
- All buttons on a page should be accessible by keyboard shortcuts and the user should be able to perform all operations using a keyboard.
- All buttons on a page should be accessible by keyboard shortcuts and the user should be able to perform all operations using a keyboard…
- All pages should have a title.
- Confirmation messages should be displayed before performing any update or delete operation.
- Hourglass should be displayed when the application is busy.
- Page text should be left-justified.
- The user should be able to select only one radio option and any combination for checkboxes .

48. What is the procedure for GUI Testing :

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

- WHAT DO YOU CHECK IN GUI TESTING?
- Check all the GUI elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
- Check you can execute the intended functionality of the application using the GUI
- Check Error Messages are displayed correctly
- Check for Clear demarcation of different sections on screen
- Check Font used in application is readable
- Check the alignment of the text is proper
- Check the Color of the font and warning messages is aesthetically pleasing

- Check that the images have good clarity
- Check that the images are properly aligned
- Check the positioning of GUI elements for different screen resolutions.

49. Write a scenario of Microwave Oven :

1. Verify that the dimensions of the oven are as per the specification provided.
2. Verify that the oven's material is optimal for its use as an oven and as per the specification.
3. Verify that the oven heats the food at the desired temperature properly.
4. Verify that the oven heats food at the desired temperature within a specified time duration.
5. Verify the ovens functioning with the maximum attainable temperature.
6. Verify the ovens functioning with minimum attainable temperature.
7. Verify that the oven's plate rotation speed is optimal and not too high to spill the food kept over it.
8. Verify that the oven's door gets closed properly.
9. Verify that the oven's door opens smoothly.
10. Verify the battery requirement of the microwave oven and check that it functions smoothly at that power.
11. Verify that the text written over the oven's body is clearly readable.
12. Verify that the digital display is clearly visible and functions correctly.
13. Verify that the temperature regulator is smooth to operate.
14. Verify that the temperature regulator works correctly.
15. Check the maximum capacity of the oven and test its functioning with that volume of food.
16. Check the oven's functionality with different kinds of food – solid, and liquid.
17. Check the oven's functionality with different food at different temperatures.
18. Verify the oven's functionality with different kinds of container material.
19. Verify that the power cord of the oven is long enough.
20. Verify that the usage instructions or user manuals have clear instructions.

50 . Write a scenario of Coffee vending Machine :

1. UI scenario – Verify that the dimension of the coffee machine is as per the specification.
2. Verify that the outer body, as well as inner part's material, is as per the specification.
3. Verify that the machine's body color as well brand is correctly visible and as per specification.
4. Verify the input mechanism for coffee ingredients-milk, water, coffee beans/powder, etc.
5. Verify that the quantity of hot water, milk, coffee powder per serving is correct.
6. Verify the power/voltage requirements of the machine.
7. Verify the effect of suddenly switching off the machine or cutting the power. The machine should stop in that situation and in power resumption, the remaining coffee should not come out of the nozzle.
8. Verify that coffee should not leak when not in operation.
9. Verify the amount of coffee served in single-serving is as per specification.
10. Verify that the digital display displays correct information.

11. Check if the machine can be switched on and off using the power buttons.
12. Check for the indicator lights when the machine is switched on-off.
13. Verify that the functioning of all the buttons work properly when pressed.
14. Verify that each button has an image/text with it, indicating the task it performs.
15. Verify that a complete quantity of coffee should get poured in a single operation, no residual coffee should be present in the nozzle.
16. Verify the mechanism to clean the system work correctly- foamer.
17. Verify that the coffee served has the same and correct temperature each time it is served by the machine.
18. Verify that the system should display an error when it runs out of ingredients.
19. Verify that pressing the coffee button multiple times leads to multiple serving of coffee.

20. Verify that there is the passage for residual/extra coffee in the machine.
21. Verify that the machine should work correctly in different climatic, moistures and temperature conditions.
22. Verify that the machine should not make too much sound when in operation.
23. Performance test – Check the amount of time the machine takes to serve a single serving of coffee.
24. Performance test – Check the performance of the machine when used continuously until the ingredients run out of the requirements.
25. Negative Test – Check the functioning of the coffee machine when two/multiple buttons are pressed simultaneously.
26. Negative Test – Check the functioning of a coffee machine with a lesser or higher voltage than required.
27. Negative Test – Check the functioning of the coffee machine if the ingredient container's capacity is exceeded.

51. Write a scenario of chair :

1. Verify that the chair is stable enough to take an average human load.
2. Check the material used in making the chair-wood, plastic etc.
3. Check if the chair's legs are level to the floor.
4. Check the usability of the chair as an office chair, normal household chair.
5. Check if there is back support in the chair.
6. Check if there is support for hands in the chair.
7. Verify the paint's type and color.
8. Verify if the chair's material is brittle or not.
9. Check if the cushion is provided with a chair or not.
10. Check the condition when washed with water or the effect of water on the chair.
11. Verify that the dimension of the chair is as per the specifications.
12. Verify that the weight of the chair is as per the specifications.
13. Check the height of the chair's seat from the floor.

52. To Create Scenario (Positive & Negative) :

Positive Test Cases for Water Bottle

Let’s see some positive test scenarios of water bottle, covering test cases on UI (User Interface), usability and functional test cases for a water bottle.

1. Verify that the dimensions of the bottle are as per the specifications.
2. Verify that the color of the bottle is as per the specifications.
3. Verify the material used in the bottle.
4. Verify the weight of the bottle is as per the specifications.
5. Verify the type of the bottle – with a lid or without a lid.
6. Check if the bottle is with a zipper or without a sipper.
7. Measure the volume of water that can be stored in the bottle and check if the volume is as specified.
8. Verify that bottle doesn’t leak when tilted or placed upside down.
9. Verify that the lid of the bottle is firmly tightened with a bottle.
10. Check the bottle’s condition with liquid of different temperatures.
11. Check the bottle's condition with different liquids – water, tea, etc.
12. Check the insulation of the bottle – time for the liquid to achieves room temperature.
13. Check the brittleness of the bottle’s material.
14. Check if the expiry date is clearly mentioned or not.
15. Verify the maximum temperature of the liquid allowed.
16. Verify the minimum temperature of the liquid allowed.

Negative Test Cases for Water Bottle

Here, we will see some negative test scenarios, in which we will cover the test cases focusing on the stress testing or robustness of the water bottle when subjected to extreme conditions.

1. Check the bottle’s condition on pouring liquid at a very high temperature, more than the permissible value.
2. Check the bottle’s condition on pouring liquid at a very low temperature, less than the permissible value.
3. Check the bottle’s condition when subjected to a very high temperature.
4. Check the bottle’s condition on pouring liquid/gas at very high pressure, more than the normal pressure.

53. Create Test Cases on Compose Mail Functionality :

https://docs.google.com/spreadsheets/d/1nT2CQdIlx1iKQaa3M1fbRkak1vbydnKtPmIInlJz9AI/edit?usp=sharing

2. Online shopping to buy product (flipkart) :

https://docs.google.com/spreadsheets/d/1VRo567vRs6fkgQHorL3HCHV01nqJdO8-Gy7A1-Er06A/edit?usp=sharing

54 . Write a Scenario of Wrist Watch :

1. Verify the type of watch – analog or digital.
2. In the case of an analog watch, check the correctness time displayed by the second, minute, and hour hand of the watch.
3. In the case of a digital watch, check if the digital display for hours, minutes, and seconds is correctly displayed.
4. Verify the material of the watch and its strap.
5. Check if the shape of the dial is as per specification.
6. Verify the dimension of the watch is as per the specification.
7. Verify the weight of the watch.
8. Check if the watch is waterproof or not.
9. Verify that the numbers in the dial are clearly visible or not.
10. Check if the watch is having a date and day display or not.

11. Verify the color of the text displayed in the watch – time, day, date, and other information.
12. Verify that the clock's time can be corrected using the key in case of an analog clock and buttons in case of a digital clock.
13. Check if the second hand of the watch makes a ticking sound or not.
14. Verify the brand of the watch and check if it's visible on the dial.
15. Check if the clock is having stopwatch, timers, and alarm functionality or not.
16. In the case of a digital watch, verify the format of the watch 12 hours or 24 hours.
17. Verify if the watch comes with any guarantee or warranty.
18. Verify if the dial has glass covering or plastic, check if the material is breakable or not.
19. Verify if the dial's glass/plastic is resistant to minor scratches or not.
20. Check the battery requirement of the watch.

55. Write a Scenario of Lift(Elevator) :

1. Verify the dimensions of the lift.
2. Verify the type of door of the lift is as per the specification.
3. Verify the type of metal used in the lift interior and exterior.
4. Verify the capacity of the lift in terms of the total weight.

5. Verify the buttons in the lift to close and open the door and numbers as per the number of floors.
6. Verify that the lift moves to the particular floor as the button of the floor is clicked.
7. Verify that the lift stops when the up/down buttons on a particular floor are pressed.
8. Verify if there is an emergency button to contact officials in case of any mishap.
9. Verify the performance of the floor – the time taken to go to a floor.
10. Verify that in case of power failure, the lift doesn't free-fall and gets halted on the particular floor.
11. Verify lifts working in case the button to open the door is pressed before reaching the destination floor.

12. Verify that in case the door is about to close and an object is placed between the doors if the doors sense the object and again open or not.
13. Verify the time duration for which the door remains open by default.
14. Verify if the lift interior is having proper air ventilation.
15. Verify lighting in the lift.
16. Verify that at no point the lift door should open while in motion.
17. Verify that in case of power loss, there should be a backup mechanism to safely get into a floor or a backup power supply.
18. Verify that in case the multiple floor number button is clicked, the lift should stop on each floor.
19. Verify that in case of capacity limit is reached users are prompted with a warning alert- audio/visual.
20. Verify that inside lift users are prompted with the current floor and direction information the lift is moving towards- audio/visual prompt.

56. Write a Scenario of Whatsapp payment :

https://docs.google.com/spreadsheets/d/1bCs_8WZz9TGI7HHO6J1sN9pS05sNrL8257q2g772JS8/edit?usp=sharing