

Module - 2 [Manual Testing]

1. What is Exploratory Testing?

- Exploratory testing is a concurrent process where
 - ◆ Test design, execution and logging happen simultaneously
 - ◆ Testing is often not recorded
 - ◆ Makes use of experience, heuristics and test patterns
 - ◆ Testing is based on a test charter that may include
 - ✧ Scope of the testing (in and out)
 - ✧ The focus of exploratory testing is more on testing as a “thinking” activity
 - ✧ A brief description of how tests will be performed
 - Expected problems
 - ◆ Is carried out in time boxed intervals

2. What is traceability matrix?

- To protect against changes you should be able to trace back from every system component to the original requirement that caused its presence.
- A software process should help you keeping the virtual table up-to-date.

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
- Boundary value analysis is a method which refines equivalence partitioning.
- Boundary value analysis generates test cases that highlight errors better than equivalence partitioning

4. What is Integration Testing?

- Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems
- Integration Testing is a level of the software testing process where individual units are combined and tested as a group

5. What determines level of risk?

- Risks should be prioritized according to their level, which is obtained by assessing the likelihood of the event occurring and the impact of that event.

6. What is Alpha testing?

- It is always performed by the developers at the software development site.
- Sometimes it is also performed by Independent Testing Team.
- Alpha Testing is not open to the market and public
- It is conducted for the software application and project.
- It is always performed in Virtual Environment.
- It is always performed within the organization.
- It is the form of Acceptance Testing.
- Alpha Testing is definitely performed and carried out at the developing organizations location with the involvement of developers.
- It comes under the category of both White Box Testing and Black Box Testing

7. What is beta testing?

- It is always performed by the customers at their own site.
- It is not performed by Independent Testing Team.
- Beta Testing is always open to the market and public.
- It is usually conducted for software product.
- It is performed in Real Time Environment.
- It is always performed outside the organization.
- It is also the form of Acceptance Testing.

8. 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.”
- Component Testing – The testing of individual software components.
- Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to
 - validate that each unit of the software performs as designed.
 - Unit testing is the first level of testing and is performed prior to Integration Testing.
 - Sometimes known as Unit Testing, Module Testing or Program Testing
 - Component can be tested in isolation – stubs/drivers may be employed
 - Unit testing frameworks, drivers, stubs and mock or fake objects are used to assist in unit testing.

9. What is functional system testing?

- A requirement that specifies a function that a system or system component must perform
- A Requirement may exist as a text document and/or a model
- There is two types of Test Approach
 - Requirement Based Functional Testing
 - Process Based Testing

10. What is Non-Functional testing?

- Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portability
- May be performed at all Test levels (not just Non Functional Systems Testing)
- Measuring the characteristics of the system/software that can be quantified on a varying scale- e.g. performance test scaling

11. What is GUI testing?

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

12. What is Adhoc testing?

- Adhoc testing is an informal testing type with an aim to break the system.
- It does not follow any test design techniques to create test cases.
- In fact it does not create test cases altogether!
- This testing is primarily performed if the knowledge of testers in the system under test is very high.
- Testers randomly test the application without any test cases or any business requirement document.
- Adhoc Testing does not follow any structured way of testing and it is randomly done on any part of application.
- Main aim of this testing is to find defects by random checking.
- Adhoc testing can be achieved with the testing technique called Error Guessing

13. What is load testing?

- It's a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under range of loads to determine at what point the system's response time degrades or fails.
- Load testing is a kind of performance testing which determines a system's performance under real-life load conditions.

14. What is 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.

15. What is white box testing and list the types of white box testing?

- Testing based on an analysis of the internal structure of the component or system.
- Structure-based testing technique is also known as 'white-box' or 'glass-box' testing technique because here the testers require knowledge of how the software is implemented, how it works

Types of Coverage:-

- Statement coverage
- Decision coverage
- Condition coverage

16. What is black box testing? What are the different black box testing techniques?

- Testing, either functional or non-functional, without reference to the internal structure of the component or system.
- Specification-based testing technique is also known as 'black-box' or input/output driven testing techniques because they view the software as a black-box with inputs and outputs.
- The testers have no knowledge of how the system or component is structured inside the box. In black-box testing the tester is concentrating on what the software does, not how it does it.

Techniques of Black Box Testing

- Equivalence partitioning
- Boundary value analysis
- Decision tables
- State transition testing
- Use-case Testing

17. Mention what are the categories of defects?

18. Mention what big-bang testing?

- In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole.
- Big Bang testing has the advantage that everything is finished before integration testing starts.

19. What is the purpose of exit criteria?

- Purpose of exit criteria is to define when we STOP testing either at the:
 - End of all testing
 - End of phase of testing

20. When should “Regression Testing” be performed?

- when the system is stable and the system or the environment changes.
- when testing bug-fix releases as part of the maintenance phase
- It should be applied at all Test Levels.
- It should be considered complete when agreed completion criteria for regression testing have been met

21. What is 7 key principals?

- ❖ Testing shows the presence of defects
- ❖ Exhaustive testing is impossible
- ❖ Early testing
- ❖ Defect clustering
- ❖ The Pesticide paradox
- ❖ Testing is context dependent
- ❖ Absence of error fallacy

■ Testing shows the presence of defects

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

■ Exhaustive testing is impossible

- Testing everything including all combinations of inputs and preconditions is not possible.
- So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts.

- So, accessing and managing risk is one of the most important activities and reason for testing in any project.
- We have learned that we cannot test everything (i.e. all combinations of inputs and pre-conditions).
- That is we must Prioritise our testing effort using a Risk Based Approach.

■ Early testing

- Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives.
- Testing activities should start as **early** as possible in the development life cycle
- These activities should be focused on defined objectives – outlined in the Test Strategy
- Remember from our Definition of Testing, that Testing doesn't start once the code has been written!

■ Defect Clustering

- A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures.
- Defects are not evenly spread in a system They are 'clustered'
- In other words, most defects found during testing are usually confined to a small number of modules
- Similarly, most operational failures of a system are usually confined to a small number of modules
- An important consideration in test prioritisation!

■ The Pesticide paradox

- If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects.
- To overcome this "pesticide paradox", the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects

■ Testing is context dependent

- Testing is basically context dependent.
- Testing is done differently in different contexts
- Different kinds of sites are tested differently.
 - For example Safety – critical software is tested differently from an e-commerce site.

■ Absence of error fallacy

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

22. Difference between QA v/s QC v/s Software Tester

S.N.	Quality Assurance	Quality Control	Testing
1	Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements.	Activities which ensure the verification of developed software with respect to documented (or not in some cases) requirements.	Activities which ensure the identification of bugs/error/defects in the Software.
2	Focuses on processes and procedures rather than conducting actual testing on the system.	Focuses on actual testing by executing Software with intent to identify bug/defect through implementation of procedures and process.	Focuses on actual testing.
3	Process oriented activities.	Product oriented activities.	Product oriented activities.
4	Preventive activities.	It is a corrective process.	It is a preventive process.
5	It is a subset of Software Test Life Cycle (STLC).	QC can be considered as the subset of Quality Assurance.	Testing is the subset of Quality Control.

23. Difference between smoke and sanity.

Smoke testing	Sanity testing
Focus on the critical functionalities of an application	Check the new functionalities or bugs fixed
The main purpose is to verify that the application work as expected before going deeper with further testing	The main purpose is to verify that main behavior of newly added features or bug fixed work well as expected
Can be done by developers or testers	Usually performed by testers
Checks the entire system	Only checks particular component of the system
Is part of Acceptance testing (to check that they meet all expectations)	Is a part of Regression testing (to check all existing features are not impacted by the changes)
Wide and shallow approach	Narrow and deep approach

24. Difference between verification v/s validation

Criteria	Verification	Validation
Definition	The process of evaluating work-products (not the actual final product) of a development phase to determine whether they meet the specified requirements for that phase.	The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements.
Objective	To ensure that the product is being built according to the requirements and design specifications. In other words, to ensure that work products meet their specified requirements.	To ensure that the product actually meets the 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.
Question	Are we building the product right?	Are we building the right product?
Evaluation Items	Plans, Requirement Specs, Design Specs, Code, Test Cases	The actual product/software.
Activities	<ul style="list-style-type: none"> Reviews Walkthroughs Inspections 	<ul style="list-style-type: none"> Testing

25. Explain types of performance testing

■ Types of Performance Testing

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

26. What is error,bug,defect and failure?

- A mistake in coding is called error, error found by tester is called defect, defect accepted by development team then it is called bug, build does not meet the requirements then it is failure

27. Difference between Priority and Severity

Features	Severity	Priority
Definition	Severity is a parameter to denote the impact of a particular defect on the software.	Priority is a parameter to decide the order in which defects should be fixed.
Purpose	Severity means how severe the defect is affecting the functionality.	Priority means how fast the defect has to be fixed.
Relation	Severity is related to the quality standard.	Priority is related to scheduling to resolve the problem.
Categories	Severity is divided into 4 categories: <ul style="list-style-type: none">• Critical• Major• Medium• Low	Priority is divided into 3 categories: <ul style="list-style-type: none">• Low• Medium• High
Who decides defects?	The testing engineer decides the severity level of the defect.	The product manager decides the priorities of defects.
Value	Its value is objective.	Its value is subjective.
Value change	Its value doesn't change from time to time.	Its value changes from time to time.
Association	It is associated with functionality or standards.	It is associated with scheduling.
Indication	It indicates the seriousness of the bug in the product functionality.	It indicates how soon the bug should be fixed.
Driving factor	It is driven by functionality	It is driven by business value.
Based On	It is based on the technical aspect of the product.	It is based on the customer's requirements.

28. What is Bug life cycle?

- As you can see from above diagram, a defect's state can be divided into Open or Closed.
- When a bug reaches one of the Closed or Terminal states, its lifecycle ends. Each state has one or more valid states to move to.
- This is to ensure that all necessary steps are taken to resolve or investigate that defect. For example, a bug should not move from
- Submitted state to resolved state without having it open.
- In a typical scenario, as soon as a bug is identified, it is logged into the bug tracking system with status as Submitted. After ascertaining the validity of the defect, it is given the "Open" Status.

29. Explain difference between Functional and NonFunctional Testing

Functional Testing	Non-functional testing
It is performed before non-functional testing.	It is performed after functional testing.
It is based on the customer's requirements.	It focusses on customer's expectations.
It is easy to define functional requirements.	It is difficult to define the requirements for non-functional testing.
It helps to validate the behavior of the application.	It helps to validate the performance of the application.
Carried out to validate software actions.	It is done to validate the performance of the software.
Functional testing is carried out using the functional specification.	This kind of testing is carried out by performance specifications
Functional testing is easy to execute by manual testing.	It's very hard to perform non-functional testing manually.
It describes what the product does.	It describes how the product works.

30. To create HLR & Test Cases of (Instagram,Facebook) first page and chat functionality

- ❖ Instagram(https://docs.google.com/spreadsheets/d/iiFe8hCF4do4VkKNp9MtIHogDYYK_G3K-9DFIOQuY4fw/edit?usp=sharing)
- ❖ Facebook(https://docs.google.com/spreadsheets/d/1qbMloV-5cXNmXBZXugk-byWFSqkRr2IbYTfC_Fyls1o/edit?usp=sharing)

31. STLC and SDLC difference

SDLC	STLC
SDLC is mainly related to software development.	STLC is mainly related to software testing.
Besides development other phases like testing is also included.	It focuses only on testing the software.
SDLC involves total six phases or steps.	STLC involves only five phases or steps.
In SDLC, more number of members (developers) are required for the whole process.	In STLC, less number of members (testers) are needed.
In SDLC, development team makes the plans and designs based on the requirements.	In STLC, testing team(Test Lead or Test Architect) makes the plans and designs.
Goal of SDLC is to complete successful development of software.	Goal of STLC is to complete successful testing of software.
It helps in developing good quality software.	It helps in making the software defects free.
SDLC phases are completed before the STLC phases.	STLC phases are performed after SDLC phases.
Post deployment support , enhancement , and update are to be included if necessary.	Regression tests are run by QA team to check deployed maintenance code and maintains test cases and automated scripts.
Creation of reusable software systems is the end result of SDLC.	A tested software system is the end result of STLC.

32. What is the difference between test scenarios , Test Cases and Test Scripts?

Test Scenario	Test Case	Test Script
Is any functionality that can be tested.	Is a set of actions executed to verify particular features or functionality.	Is a set of instructions to test an app automatically.
Is derived from test artifacts like Business Requirement Specification (BRS) and Software Requirement Specification (SRS).	Is mostly derived from test scenarios.	Is mostly derived from test cases.
Helps test the end-to-end functionality in an Agile way.	Helps in exhaustive testing of an app.	Helps to test specific things repeatedly.
Is more focused on what to test.	Is focused on what to test and how to test.	Is focused on the expected result.
Takes less time and fewer resources to create.	Requires more resources and time.	Requires less time for testing but more resources for scripts creating and updating.
Includes an end-to-end functionality to be tested.	Includes test steps, data, expected results for testing.	Includes different commands to develop a script.
The main task is to check the full functionality of a software application.	The main task is to verify compliance with the applicable standards, guidelines, and customer requirements.	The main task is to verify that nothing is skipped, and the results are true as the desired testing plan.
Allows quickly assessing the testing scope.	Allows detecting errors and defects.	Allows carrying out an automatic execution of test cases.

33. Explain What is test plan?What is the information that should be covered

- *A document describing the scope, approach, resources and schedule*
- *of intended test activities* Determining the scope and risks, and identifying the objectives of testing.
- Defining the overall approach of testing (the test strategy), including the definition of the test levels and entry and exit criteria.
- Integrating and coordinating the testing activities into the software life cycle

■ activities:

- acquisition, supply, development, operation and maintenance.
- Making decisions about what to test, what roles will perform the test activities, how the test activities should be done, and how the test results will be evaluated?
- Scheduling test analysis and design activities.
- Scheduling test implementation, execution and evaluation.
- Assigning resources for the different activities defined
 - Defining the amount, level of detail, structure and templates for the test documentation.

34. What is priority?

- The order in which bugs are addressed, based on how soon they need to be fixed:

P0 (Critical)

These bugs have a huge impact on the business and are addressed immediately, regardless of their severity. For example, an e-commerce site defect that prevents users from checking out.

P1 (High)

These bugs affect the business and are addressed quickly. For example, a search feature that returns results slowly but users can still perform other functions.

P2 (Medium)

These bugs can be ignored for a while because they don't significantly impact the software's functionality.

P3 (Low)

These bugs are addressed only after higher priority bugs are resolved.

35. What is severity?

- Severity is defined as the extent to which a particular defect can create an impact on the software. Severity is a parameter to denote the implication and the impact of the defect on the functionality of the software.
 - A higher effect of the bug on system functionality will lead to a higher severity level.
 - A QA engineer determines the severity level of a bug.

Types of Severity:

- ◆ **Critical:** This severity level implies that the process has been completely shut off and no further action can be taken.
- ◆ **Major:** This is a significant flaw that causes the system to fail. However, certain parts of the system remain functional.
- ◆ **Medium:** This flaw results in unfavorable behavior but the system remains functioning.
- ◆ **Low:** This type of flaw won't cause any major breakdown in the system.

36. Bug categories are...




Bug Category: Security, Database, Functionality (Critical/General), UI

37. Advantages of Bugzilla.

- **Deadlines:** To fix the bugs, deadlines can be established.
- **Types:** It reports in a variety of formats and types.
- **Request System:** You can use the 'request system' provided by Bugzilla to ask other users to evaluate codes, provide information and other things.
- **Flexible:** Bugzilla is quite flexible, so you can modify it to fit your unique process and requirements.
- **Bug tracking tool:** Bugzilla is extremely good at monitoring and handling bugs and issues.

38. What are the different methodologies in agile Development method?

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

Authentication	Authorization
The process of verifying the identity of a user, device, or system.	The process of granting or denying access to a specific resource or action.
Determines who the user is. Examples: login credentials, biometric data.	Determines what the user is allowed to do. Examples: permissions, roles, access controls.
Authentication happens before authorization. 	Authorization happens after authentication. 
Authentication is a requirement for authorization.	Without authentication, authorization cannot occur.
The authentication credentials can be changed in part as and when required by the user.	The authorization permissions cannot be changed by the user as these are granted by the owner of the system and only he/she has the access to change it.
The user authentication is visible at user end.	The user authorization is not visible at the user end. 
The user authentication is identified with a username, password, face recognition, fingerprints, etc.	The user authorization is carried out through the access rights to resources by using roles that have been pre-defined.

Some common web testing problems include: [\[1\]](#)

Weak password policies

Weak passwords, such as those that are not complex or long enough, can make it easier for attackers to compromise user accounts.

Excessive data exposure

APIs can expose sensitive data if developers don't take enough care of data security.

Insecure value comparisons

Insecure direct object references can leave an internal object exposed to the user, making it vulnerable to attack.

User logout

Failing to destroy the server-side session can keep the session or token alive after the user logs out.

Inadequate API security measures

APIs without proper authentication, encryption, and access controls are vulnerable to unauthorized access and data breaches.

Authentication is the process of verifying a user's identity, while authorization is the process of granting the user permission to access different levels of information and perform specific functions.

40. To create HLR & TestCase of WebBased(WhatsappWeb)

- <https://docs.google.com/spreadsheets/d/1gPo-AqwZTG4V8-W65n8QEOkm2zqfJrCojzkxhvvMlBA/edit?usp=sharing>

41. Test Scenario of Pen.

- Verify the type of pen, whether it is a ballpoint pen, ink pen, or gel pen.
- Verify that the user is able to write clearly over different types of papers.
- 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.
- Verify if the pen is with a cap or without a cap.
- Verify the color of the ink on the pen.
- Check the odor of the pen's ink on writing over a surface.
- Verify the surfaces over which the pen is able to write smoothly apart from paper e.g. cardboard, rubber surface, etc.
- Verify that the text written by the pen should have consistent ink flow without leaving any blob.
- Check that the pen's ink should not leak in case it is tilted upside down.

- Verify if the pen's ink should not leak at higher altitudes.
- Verify if the text written by the pen is erasable or not.
- Check the functioning of the pen by applying normal pressure during writing.
- Verify the strength of the pen's outer body. It should not be easily breakable.
- Verify that text written by pen should not get faded before a certain time as mentioned in the specification.
- Check if the text written by the pen is waterproof or not.
- Verify that the user is able to write normally by tilting the pen at a certain angle instead of keeping it straight while writing.
- Check the grip of the pen, and whether it provides adequate friction for the user to comfortably grip the pen.
- Verify if the pen can support multiple refills or not.
- In the case of an ink pen, verify that the user is able to refill the pen with all the supported ink types.
- For ink pens, verify that the mechanism to refill the pen is easy to operate.
- In the case of a ballpoint pen, verify the size of the tip.
- In the case of a ball and gel pen, verify that the user can change the refill of the pen easily.

42. Test Scenario for Door.

- Verify if the door is single door or bi-folded door.
- Check if the door opens inwards or outwards.
- Verify that the dimension of the doors are as per the specifications.
- Verify that the material used in the door body and its parts is as per the specifications.
- Verify that color of the door is as specified.
- Verify if the door is sliding door or rotating door.
- Check the position, quality and strength of hinges.
- Check the type of locks in the door.

- Check the number of locks in the door interior side or exterior side.
- Verify if the door is having peek-hole or not.
- Verify if the door is having stopper or not.
- Verify if the door closes automatically or not – spring mechanism.
- Verify if the door makes noise when opened or closed.
- Check the door condition when used extensively with water.
- Check the door condition in different climatic conditions- temperature, humidity etc.
- Check the amount of force- pull or push required to open or close the door.

43. Test Scenario of ATM.

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

- 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.
- Check that the pin is displayed in masked form when entered.
- Verify that the user is presented with different account type options like- saving, current, etc.
- Verify that the user is allowed to get account details like available balance.
- Check that the correct amount of money gets withdrawn as entered by the user for cash withdrawal.
- Verify that the user is only allowed to enter the amount in multiple denominations as per the specifications.
- Verify that the user is prompted to enter the amount again in case the amount entered is less than the minimum amount configured.
- Check that the user cannot withdraw more amount than the total available balance and a proper message should be displayed.
- Verify that the user is provided the option to get the transaction details in printed form.
- Verify that the user's session timeout is maintained.
- Check that the user is not allowed to exceed one transaction limit amount.
- Verify that the user is not allowed to exceed the one-day transaction limit amount.
- Verify that the user is allowed to do only one transaction per pin request.
- Check that in case the ATM machine runs out of money, a proper message is displayed to the user.
- 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.

- 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.
- Check that the user is not allowed to proceed with the expired ATM card and that a proper error message gets displayed.
- 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.

44. When to use Usability Testing?

1. During Early Prototyping or Design Phase

- Why: To gather feedback on wireframes, mockups, or prototypes before significant development efforts.
- Benefits:
 - Identify usability issues early.
 - Ensure the design aligns with user expectations.

2. Before Major Development Milestones

- Why: To validate the usability of newly developed features or modules.
- Benefits:
 - Avoid costly redesigns later.
 - Confirm that the implementation matches user needs.

3. During Pre-release Testing

- Why: To ensure the final product is easy to use before it goes live.
- Benefits:
 - Catch any usability issues that may have been overlooked.
 - Test the product in its complete form with real users.

4. Post-launch Enhancements

- Why: To evaluate how users interact with the product after release and gather feedback for future updates.

- Benefits:
 - Improve existing features.
 - Address user pain points based on real-world usage.

5. When Introducing New Features

- Why: To ensure the new feature integrates seamlessly with the existing user experience.
- Benefits:
 - Prevent disruption to user workflows.
 - Validate that new functionality adds value.

6. For Continuous Improvement

- Why: To periodically test and refine the product to maintain a competitive edge and meet evolving user needs.
 - Benefits:
 - Ensure long-term satisfaction and retention.

45. What is the procedure for 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 resolution.

46. Write Scenario of Microwave Owen.

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

47. Write a Scenario of Coffee vending machine.

- UI scenario – Verify that the dimension of the coffee machine is as per the specification.
- Verify that outer body, as well as inner part's material, is as per the specification.
- Verify that the machine's body color as well brand is correctly visible and as per specification.
- Verify the input mechanism for coffee ingredients-milk, water, coffee beans/powder, etc.
- Verify that the quantity of hot water, milk, coffee powder per serving is correct.
- Verify the power/voltage requirements of the machine.
- 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 get come out of the nozzle.
- Verify that coffee should not leak when not in operation.
- Verify the amount of coffee served in single-serving is as per specification.
- Verify that the digital display displays correct information.
- Check if the machine can be switched on and off using the power buttons.
- Check for the indicator lights when the machine is switched on-off.
- Verify that the functioning of all the buttons work properly when pressed.
- Verify that each button has an image/text with it, indicating the task it performs.

- Verify that complete quantity of coffee should get poured in a single operation, no residual coffee should be present in the nozzle.
- Verify the mechanism to clean the system work correctly-foamer.
- Verify that the coffee served has the same and correct temperature each time it is served by the machine.
- Verify that system should display an error when it runs out of ingredients.
- Verify that pressing the coffee button multiple times leads to multiple serving of coffee.
- Verify that there is the passage for residual/extra coffee in the machine.
- Verify that machine should work correctly in different climatic, moistures and temperature conditions.
- Verify that machine should not make too much sound when in operation.
- Performance test – Check the amount of time the machine takes to serve a single serving of coffee.
- Performance test – Check the performance of the machine when used continuously until the ingredients run out of the requirements.
- Negative Test – Check the functioning of the coffee machine when two/multiple buttons are pressed simultaneously.
- Negative Test – Check the functioning of coffee machine with a lesser or higher voltage than required.
- Negative Test – Check the functioning of the coffee machine if the ingredient container's capacity is exceeded.

48. Write a Scenario of Chair.

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

49. Create test cases on compose mail functionality.

- https://docs.google.com/spreadsheets/d/1Gr1k2l_yG5CKpM5tMAxlvnXuaoofBHAapr87kJ-P7rE/edit?usp=sharing

50. Write a Scenario of wrist watch.

- Verify the type of watch – analog or digital.
- In the case of an analog watch, check the correctness time displayed by the second, minute, and hour hand of the watch.
- In the case of a digital watch, check the digital display for hours, minutes, and seconds is correctly displayed.
- Verify the material of the watch and its strap.
- Check if the shape of the dial is as per specification.
- Verify the dimension of the watch is as per the specification.
- Verify the weight of the watch.
- Check if the watch is waterproof or not.
- Verify that the numbers in the dial are clearly visible or not.
- Check if the watch is having a date and day display or not.
- Verify the color of the text displayed in the watch – time, day, date, and other information.
- Verify that clock's time can be corrected using the key in case of an analog clock and buttons in case of a digital clock.
- Check if the second hand of the watch makes ticking sound or not.
- Verify if the brand of the watch and check if its visible in the dial.
- Check if the clock is having stopwatch, timers, and alarm functionality or not.
- In the case of a digital watch, verify the format of the watch 12 hours or 24 hours.
- Verify if the watch comes with any guarantee or warranty.
- Verify if the dial has glass covering or plastic, check if the material is breakable or not.
- Verify if the dial's glass/plastic is resistant to minor scratches or not.
- Check the battery requirement of the watch.

51. Write a Scenario of Lift(elevator)

- Verify the dimensions of the lift.
- Verify the type of door of the lift is as per the specification.
- Verify the type of metal used in the lift interior and exterior.
- Verify the capacity of the lift in terms of the total weight.
- Verify the buttons in the lift to close and open the door and numbers as per the number of floors.
- Verify that the lift moves to the particular floor as the button of the floor is clicked.
- Verify that the lift stops when the up/down buttons on a particular floor are pressed.
- Verify if there is an emergency button to contact officials in case of any mishap.
- Verify the performance of the floor – the time taken to go to a floor.
- Verify that in case of power failure, the lift doesn't free-fall and gets halted on the particular floor.
- Verify lifts working in case the button to open the door is pressed before reaching the destination floor.
- 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.
- Verify the time duration for which the door remains open by default.
- Verify if the lift interior is having proper air ventilation.
- Verify lighting in the lift.
- Verify that at no point the lift door should open while in motion.
- Verify that in case of power loss, there should be a backup mechanism to safely get into a floor or a backup power supply.

- Verify that in case the multiple floor number button is clicked, the lift should stop on each floor.
- Verify that in case of capacity limit is reached users are prompted with a warning alert- audio/visual.
- Verify that inside lift users are prompted with the current floor and direction information the lift is moving towards- audio/visual prompt.

52. Create Test Cases for online shopping to Buy Product (Flipkart)

- <https://docs.google.com/spreadsheets/d/18x3LWInIVsOWOBoKklcaCeIuCf4T1wZMfeA5ynQxlfA/edit?usp=sharing>