What is Exploratory Testing?

Exploratory Testing is a software testing method where testers learn, design test cases, and test the application at the same time.
There is no fixed test plan; instead, testers explore the software freely based on their understanding.

In this testing, the tester uses creativity, experience, and intuition to find defects. It is useful when requirements are unclear, or when quick feedback is needed.

What is a traceability matrix?

A Traceability Matrix (RTM) is a document that maps and traces requirements with test cases.
Its purpose is to ensure every requirement is tested and nothing is missed.

What is Boundary value testing?

Boundary Value Testing (BVA) is a technique where testing is done on the border values of input fields.
Most defects occur at the edges of input ranges, so testing at boundary values helps detect errors quickly.

For example, if age allowed is 18 to 60, then test values are:
17, 18, 60, 61.

It reduces time and increases defect detection, especially in numeric fields.

What is Equivalence partitioning testing?

Equivalence Partitioning divides input data into logical groups called partitions.
Data inside the same group behaves similarly, so testing one value from each group is enough.

For example, if valid age is 18–60:

- Invalid group 1: below 18
- Valid group 2: between 18 and 60
- Invalid group 3: above 60

This method reduces test cases and ensures wide coverage.

Integration Testing is the process of testing how two or more modules interact with each other.
Even if each module works well alone, they may fail when combined.

Goals:

- Find issues in module interfaces
- Check data flow between modules
- Ensure modules work properly as a unit

Examples:
Payment module + Order module,
Login module + Dashboard module.

What determines the level of risk?

Risk level depends mainly on two things:

1. Impact – how serious the damage will be
2. Probability – how likely it is to happen

Formula:
Risk = Impact × Probability

High impact + high probability = High risk
Low impact + low probability = Low risk

What is Alpha Testing?

Alpha Testing is the first phase of user testing done by internal employees—developers, testers, and product teams.
It is performed before the software is released to users.

Characteristics:

- Done in development environment
- Controlled testing
- Helps find major issues early
- Feedback comes from internal staff

What is Beta Testing?

Beta Testing is the second phase where the software is given to real users to use in a real environment.
It helps collect user feedback, performance issues, and usability problems.

Characteristics:

- Done in customer environment
- Identifies issues not found in alpha testing
- Checks customer satisfaction
- Improves product quality before final release

What is Component Testing?

Component Testing (Unit Testing) checks individual modules or components independently.
It ensures that each component works as expected.

It mainly focuses on:

- Correctness of logic
- Input–output behavior
- Boundary conditions
- Error handling

Usually done by developers before integration.

What is Functional System Testing?

Functional System Testing checks the entire system as a whole to ensure all functions work according to the requirement.
It is performed after integration testing is completed.

It verifies:

- All features
- Workflows
- User interactions
- Input validations

Goal is to ensure the system meets customer needs.

What is Non-Functional Testing?

Non-Functional Testing checks the quality aspects of the software like:

- Performance
- Load
- Security
- Usability
- Reliability
- Compatibility

It tests how well the system works, not what the system does.

Example:
Checking website speed, response time, and security.

What is GUI Testing?

GUI Testing ensures that the Graphical User Interface is working properly.
GUI includes everything the user sees:

- Buttons
  Textboxes
- Dropdowns
- Icons
- Colors
- Layout

GUI testing checks if the interface is user-friendly, consistent, and functioning without errors.

What is Adhoc Testing?

Adhoc Testing is unplanned testing without any formal test cases.
The tester explores the application freely based on experience.

It helps find unexpected defects quickly, especially when there is less time.

What is Load Testing?

Load Testing checks how the system behaves under expected amount of load (expected number of users).

Purpose:

- To check response time
- To ensure the system can handle expected traffic
- To find performance bottlenecks

Example: 500 users accessing a website at the same time.

What is Stress Testing?

Stress Testing checks how the system behaves under extreme load, more than its capacity.

Purpose:

- To identify breaking point
- To check stability after crash
- To test recovery behavior

Example: Testing with 5,000 users when system limit is 1,000.

What is White Box Testing? Types?

White Box Testing tests the internal structure, logic, and code of the software.
The tester must know programming.

Types:

- Statement Coverage – checks every line
- Branch Coverage – checks each decision (true/false paths)
- Condition Coverage – checks all logical conditions
- Path Coverage – checks all possible paths
- Loop Testing – checks loop functioning

What is Black Box Testing?

Black Box Testing tests the software without knowing the internal code.
Tester only sees input and output.

It checks:

- Functional requirements
- UI
- Behavior
- Valid/Invalid inputs

Used by testers to validate user functionalities.

Techniques:

- Equivalence Partitioning
  Boundary Value Analysis
- Decision Table Testing
- State Transition Testing
- Use Case Testing
- Error Guessing

Mention what are the categories of defects?

Defects can be categorized based on severity:

- Critical – system crash
- High – major feature broken
- Medium – partial function issue
- Low – minor UI issues

Or Based on type:
Functional, UI, Performance, Security, Compatibility, Logical defects.

What is Big Bang Testing?

Big Bang Testing is a method where all modules are integrated at once and the entire system is tested together.
It is simple but risky because:

- Hard to find the root cause of defects
- No step-by-step integration
- Late detection of major issues

Purpose of Exit Criteria

Exit criteria define when testing can stop.
It ensures the software is ready for release.

Examples:

- All test cases executed
- No high/critical bugs open
- Required test coverage achieved
- Deadlines met

Without exit criteria, testing may continue endlessly.

When should Regression Testing be performed?

Regression Testing is performed after any change in the software, such as:

- Bug fix

- New feature
- Code modification
- Enhancement

Its purpose is to check that old functionality still works after changes.

What is 7 key principles? Explain in detail?

1. Testing shows defects
   Testing can find defects but cannot prove the software is 100% defect-free.

2. Exhaustive testing is impossible
   You cannot test all inputs or paths; only selective testing is possible.

3. Test early
   Testing early reduces cost and effort.

4. Defects cluster together
   Most defects occur in a few modules.

5. Pesticide paradox
   Running the same tests repeatedly finds fewer new bugs, so tests must be updated.

6. Testing depends on context
   Testing approach differs based on application type.

7. Absence-of-error is a myth
   Even if no defects are found, it doesn't mean the system meets user needs.

Difference between QA vs QC vs Tester

Feature	QA (Quality Assurance)	QC (Quality Control)	Tester
Definition	QA ensures that **processes** used to create the product are correct.	QC ensures that the **final product** meets the required quality.	A Tester is the person who **executes tests** on the product.
Focus	**Process-focused** (prevents defects).	**Product-focused** (finds defects).	**Testing-focused**.

Goal	Improve and maintain the development **process**.	Identify defects in the **product**.	Find bugs and verify functionality.
When it Happens	Before development and during development.	After development.	During testing phase.
Nature	**Proactive** – stops defects from entering.	**Reactive** – detects defects after product is built.	Executes test cases, reports defects.
Activities	Process audits, training, standards, reviews.	Inspection, testing, test result evaluation.	Test case execution, retesting, reporting.
Responsibility	QA team / management.	QC team.	Tester/ Testing team.

Difference between Smoke and Sanity Testing

Feature	**Smoke Testing**	**Sanity Testing**
Purpose	To check if the **build is stable** for further testing.	To check if **specific functions** work after a bug fix or update.
Type	Shallow and wide.	Narrow and deep.
When Done	After new build release.	After receiving a minor change or bug fix.
Covers	Major and critical features only.	Only the changed or related features.

Goal	Ensure application doesn't crash at basic level.	Ensure recent changes are working correctly.
Documentation	Usually documented.	Usually not documented.
Performed By	Testers.	Testers.

Difference between Verification and Validation

Feature	**Verification**	**Validation**
Meaning	Check whether we are **building the product right**.	Checks whether we are **building the right product**.
Focus	Processes, documents, and designs.	Actual software behavior and functionality.
Execution	No code execution.	Requires code execution.
Activities	Reviews, walkthroughs, inspections.	Functional testing, system testing, UAT.
Performed By	QA team, developers, analysts.	Testing team and end users.
When It Happens	During development phases.	After development is completed.
Objective	To prevent defects.	To check if software meets user requirements.

Types of Performance Testing

- Load Testing
- Stress Testing
- Scalability Testing
- Spike Testing
- Endurance (Soak) Testing
- Volume Testing

Each type checks different aspects of performance like speed, stability, and handling ability.

What is Error, Defect, Bug, Failure?

- Error: A mistake made by a developer.
- Defect: Error found during testing.
- Bug: Defect accepted by developers.
- Failure: When a defect reaches users and causes malfunction.

Difference between Priority and Severity

Feature	Priority	Severity
Meaning	Priority tells how fast a defect should be fixed.	Severity tells how serious the defect is for the system.
Focus	Business importance & urgency.	Technical impact & damage to the system.
Decided By	Product Manager / Business Team / Client.	Tester (based on testing impact).

Represents	Importance of fixing the bug immediately.	How much the bug affects system functionality.
Fixing Order	High priority bugs are fixed first even if severity is low.	High severity bugs may or may not be fixed first depending on priority.
Type	Can be: High, Medium, Low.	Can be: Critical, High, Medium, Low.
Example	High Priority + Low Severity: A spelling mistake on the homepage ("Gogle" instead of "Google"). Looks bad, must fix soon.	High Severity + Low Priority: System crashes only when clicking a rarely used admin report. Serious but used rarely.

What is Bug Life Cycle?

Stages of a bug from opening to closing:

1. New – reported
2. Assigned – given to developer
3. Open – developer starts working
4. Fixed – issue solved
5. Retest – tester checks again
   Verified – passed retesting
6. Closed – no issue
7. Reopened – if defect persists

Explain the difference between Functional testing and Non Functional testing

Feature	Functional Testing	Non-Functional Testing

Meaning	Tests **what the system does**. It checks whether the software functions according to the requirements.	Tests **how the system works**. It checks performance, speed, usability, reliability, etc.
Focus Area	Checks features, actions, and business logic.	Checks quality attributes like performance, security, speed.
Objective	To ensure the application performs **correct functions**.	To ensure the application performs well under **different conditions**.
Execution Method	Gives input → checks expected output.	Evaluates system behavior under load, stress, usability tests.
Examples	Login working or not, form submission, search, payment process.	Page loading speed, number of users system supports, security tests, usability tests.
Testing Type	Includes: Smoke, Sanity, Regression, Integration, System Testing, UAT.	Includes: Performance, Load, Stress, Scalability, Security, Usability testing.
Importance	Ensures required functions are correct and usable.	Ensures system quality, stability, user experience, and performance.

Performed By	Testing team (functional testers).	Performance testers, security testers, and specialized teams.

What isthe difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?

Feature	SDLC (Software Development Life Cycle)	STLC (Software Testing Life Cycle)
Meaning	SDLC is the entire process of developing software, from planning to deployment.	STLC is the process of testing the software, from test planning to test closure.
Focus	Focuses on building the product.	Focuses on verifying and validating the product.
Activities	Requirement gathering, design, coding, testing, deployment, maintenance.	Requirement analysis, test planning, test design, test execution, defect reporting, closure.
Performed By	Business analysts, designers, developers, testers.	Testing team only.

Output	Working software product.	Tested and quality-assured product.
Objective	To develop high-quality software based on user requirements.	To ensure the developed software is correct, bug-free, and meets requirements.

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

Term	Meaning	Example
Test Scenario	A high-level description of what to test. It tells the overall idea.	“Verify login functionality.”
Test Case	A detailed document that includes steps, input, expected result for a scenario.	Step 1: Enter username… Expected Result: Login successful.
Test Script	A set of automated test instructions written in a scripting language or automation tool.	Selenium/Python script for login test.

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

A **Test Plan** is a formal document that describes **how testing will be carried out**.
It provides a complete strategy, resources, schedule, scope, and objectives for testing the software.

A Test Plan ensures

- Testing is organized
- All team members follow the same process
- Risks are handled properly
- Quality goals are achieved

A good Test Plan should cover the following information:

1. **Test Plan ID** – unique number for identification
2. **Introduction** – purpose of the test plan
3. **Scope of Testing**
    - **In-scope**: what will be tested

    - **Out-of-scope**: what will NOT be tested
4. **Test Objectives** – goals like finding defects, verifying requirements
5. **Test Strategy / Approach**
    - Type of testing (functional, regression, system, smoke, etc.)
6. **Test Environment**
    - Hardware, software, network setup
7. **Roles and Responsibilities**
    - Who will test what?
8. **Entry & Exit Criteria**
    - When to start and stop testing
9. **Test Deliverables**
    - Test cases, reports, defect logs
10. **Schedule / Timeline**
    - Testing start and end dates
11. **Risk and Mitigation Plan**
    - What could go wrong and how to handle it
12. **Tools to be used**
    - JIRA, Selenium, TestRail, etc.
13. **Approvals**
    - Sign-off from managers or leads

What is Priority?

Priority means **how fast the bug should be fixed**.
It is decided by the **Product Manager / Business team**.
It tells the **order of fixing bugs**.

Example:
• "Login button not aligned" = Low Priority
• "App not opening" = High Priority

What is Severity?

Severity means **how big the impact of the bug is on the system**.
It is decided by the **Tester**.
It tells **how serious the bug is for functionality**.

Example:
• Divide calculation giving wrong result = High Severity
• Spelling mistake = Low Severity

Bug Categories Are…

Common bug categories:

1. **Functional Bugs** – features not working
2. **Performance Bugs** – slow speed, crash
3. **Compatibility Bugs** – not working in some devices/browser
4. **UI/UX Bugs** – design, alignment issues
5. **Security Bugs** – login bypass, data leak

6. **Regression Bugs** – old feature broken after new change
7. **Logical Bugs** – wrong output, wrong calculation

Advantages of Bugzilla

1. Free and open-source tool
2. Easy bug tracking and workflow
3. Allows file attachment (screenshots/logs)
4. Email notifications for bug updates
5. Search + Filter options for all bugs
6. Supports multiple users and projects
7. Good reporting (bug history, status, changes)

Difference Between Priority and Severity

Feature	Priority	Severity
Meaning	Urgency to fix bug	Impact of bug
Decided by	Business/Manager	Tester
Focus	Business needs	Technical seriousness
Example	App crash in rare case = Low Priority	App crash anytime = High Severity

Different Methodologies in Agile Development

Common Agile methodologies are:

1. **Scrum** – Work in sprints (2–4 weeks), daily stand-ups
2. **Kanban** – Continuous flow, task boards
3. **XP (Extreme Programming)** – Frequent releases, pair programming, TDD
4. **Lean** – Reduce waste, faster delivery
5. **Crystal** – Different approaches based on team size
6. **DSDM** – Rapid development with user involvement

Difference Between Authentication & Authorization (Web Testing)

Authentication

• Checking **WHO** the user is
 • Example: Login with username and password

Authorization

• Checking **WHAT** user is allowed to access
 • Example: Normal user cannot access Admin page

Simple Line:
Authentication = “Are you a valid user?”
Authorization = “What can you do inside the system?”

Common Problems Faced in Web Testing

1. Browser Compatibility issues – website not working same on Chrome/Firefox
2. Responsive Issues – layout breaks on mobile
3. Slow Loading / Performance
4. Broken Links
5. Security issues – login bypass, weak password
6. Form validation issues
7. Session expire issues
8. Incorrect alignment/UI mismatch
9. Different behaviors due to cookies/cache

Write a scenario of only Whatsapp chat messages

- Verify that the user can open a chat window by selecting a contact from the chat list.
- Verify that the user can type a message in the text input field and send it successfully.
- Verify that sent messages appear in the chat window with the correct timestamp.
- Verify that received messages from the contact appear in real-time in the chat window.
- Verify that the message delivery status is displayed correctly with ticks (one tick = sent, two ticks = delivered, blue ticks = read).
- Verify that the user can send images, videos, and documents to the contact.
- Verify that the user can delete a message using “Delete for me” or “Delete for everyone” options.
- Verify that the chat window displays all sent and received messages in chronological order.
- Verify that the user can scroll through the chat to view previous messages.
- Verify that the chat list shows contact’s DP, name, and last message preview correctly.
- Verify that clicking on a message and selecting “Message Info” displays delivery and read time.

Write a Scenario of Pen

1. Verify that the length and the diameter of the pen are as per the specifications.
2. Verify the outer body material of the pen. Check if it is metallic, plastic, or any other material specified in the requirement specifications.
3. Check the color of the outer body of the pen. It should be as per the specifications.

4. Verify that the brand name and/or logo of the company creating the pen should be clearly visible.
5. Verify that any information displayed on the pen should be legible and clearly visible.
6. Verify the type of pen, whether it is a ballpoint pen, ink pen, or gel pen.
7. Verify that the user is able to write clearly over different types of papers.
8. 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.
9. Verify if the pen is with a cap or without a cap.
10. Verify the color of the ink on the pen.
11. Check the odor of the pen's ink on writing over a surface.
12. Verify the pen's performance on different paper textures (smooth, rough, glossy, etc.) to ensure consistent ink dispersion and grip.
13. Verify the surfaces over which the pen is able to write smoothly apart from paper e.g. cardboard, rubber surface, etc.
14. Verify that the text written by the pen should have consistent ink flow without leaving any blob.
15. Check that the pen's ink should not leak in case it is tilted upside down.
16. Verify if the pen's ink should not leak at higher altitudes.
17. Verify if the text written by the pen is erasable or not.
18. Check the functioning of the pen by applying normal pressure during writing.
19. Verify the strength of the pen's outer body. It should not be easily breakable.
20. Verify that text written by pen should not get faded before a certain time as mentioned in the specification.
21. Check if the text written by the pen is waterproof or not.
22. Verify that the user is able to write normally by tilting the pen at a certain angle instead of keeping it straight while writing.
23. Check the grip of the pen, and whether it provides adequate friction for the user to comfortably grip the pen.
24. Verify if the pen can support multiple refills or not.
25. In the case of an ink pen, verify that the user is able to refill the pen with all the supported ink types.
26. For ink pens, verify that the mechanism to refill the pen is easy to operate.
27. In the case of a ballpoint pen, verify the size of the tip.
28. In the case of a ball and gel pen, verify that the user can change the refill of the pen easily.

Write a Scenario of Pen Stand

Verify that the pen stand can hold multiple pens without tipping over.
Verify that the pen stand is stable on a flat surface.
Verify that the user can easily insert and remove pens from the stand.
Verify that the pen stand does not scratch or damage the pens.
Verify that the pen stand can hold pens of different sizes (ballpoint, gel, marker).
Verify that the compartments are of adequate size for storing pens and pencils.
Verify that the pen stand remains upright even when all compartments are filled.
Verify that the material of the pen stand is durable and does not break under normal use.
Verify that the pen stand can be cleaned easily without damage.
Verify that the pen stand does not slip if placed on a smooth surface.

Write a Scenario of 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.

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.

When to use Usability Testing?

Usability Testing is performed to **check how user-friendly and easy to use an application or system is**.

When to use it:

1. **Before releasing the software to users** – to ensure a good user experience.
2. **When a new interface or feature is added** – to verify it is easy to navigate.
3. **When the application is complex** – to make sure users can perform tasks efficiently.
4. **For mobile apps, websites, or any interactive system** – to check layout, readability, and workflow.
5. **During iterative design** – to identify and fix usability problems early.

Goal: Make the system intuitive, reduce user errors, and improve satisfaction.

What is the procedure for GUI Testing?

GUI Testing checks the Graphical User Interface of an application to ensure it works correctly and looks good.

Steps / Procedure for GUI Testing:

1. Understand requirements and design
    - Review GUI specifications, layout, colors, fonts, buttons, menus, and forms.
2. Prepare test cases
    - Create scenarios for buttons, links, text boxes, drop-downs, images, pop-ups, and error messages.
3. Set up test environment
    - Ensure the application runs on the required OS, browser, or device.
4. Execute GUI tests
    - Check layout, alignment, font sizes, colors, labels, images, and icons.
    - Verify all buttons, links, and menus are clickable and work as expected.
    - Test input fields for valid/invalid inputs and error messages.
    - Test window resizing, scrolling, and responsiveness.
5. Record results
    - Note any mismatch, broken links, missing icons, or alignment issues.
6. Report defects
    - Log all GUI defects in the bug tracking tool for fixing.
7. Re-test after fixes

- Verify the defects are fixed and GUI is consistent.

Write a scenario of Microwave Owen

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 function's 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 instruction or user manuals have clear instructions.

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 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 get 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 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 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 machine should work correctly in different climatic, moistures and temperature conditions.
22. Verify that 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.

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 leg 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 cushion is provided with chair or not.
10. Check the condition when washed with water or effect of water on chair.
11. Verify that the dimension of 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

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 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 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 ticking sound or not.
14. Verify if the brand of the watch and check if its visible in 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.

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.

Write a Scenario of Whatsapp payment

- Verify that the user can open WhatsApp and access the Payments option from the menu.
- Verify that the user can link a bank account to WhatsApp Payments successfully.
- Verify that the user receives an OTP or verification code from the bank and can verify the account.
- Verify that the user can send money to a contact from the WhatsApp contact list.
- Verify that the user can enter the correct amount and add a note (optional) before sending money.
- Verify that the user receives a confirmation message after successfully sending money.
- Verify that the receiver gets the payment in their linked account instantly.
- Verify that the user can request money from a contact using WhatsApp Payments.
- Verify that the user can view payment history including sent and received payments.
- Verify that the user can delete or cancel a pending payment if allowed.
- Verify that appropriate error messages appear for:
- Insufficient balance
- Wrong bank account
- Exceeding daily transaction limit
- Verify that the user can update or remove the linked bank account if required.