

- **What is Exploratory Testing?**

- Exploratory Testing means testing a software by exploring it freely, without following written test steps, to find problems while you learn how it works.

- **What is Traceability matrix?**

- A traceability matrix is a table that links requirements to test cases to make sure every requirement is tested and nothing is missed.

- **What is Boundary value testing?**

- It is a testing method where you check the edge values (just at, below, and above the limits) because bugs often happen at the boundaries of input ranges.

- **What is Equivalence partitioning testing?**

- It is a testing method where you divide inputs into groups (partitions) that should behave the same, and then test just one value from each group instead of testing everything.

- **What is Integration testing?**

- Integration Testing is a type of software testing where two or more modules are combined and tested together to check if they work correctly as a group.

- **What determines the level of risk?**

- The level of risk is determined by two main factors:

1. Probability (Likelihood) – How likely it is that a problem or failure will happen.
2. Impact (Severity) – How bad the damage or effect will be if the problem happens.

Risk Level = Likelihood × Impact

- **What is Alpha testing?**

- Alpha Testing is a type of software testing done by the internal team (developers or testers) before releasing the product to real users.

- **What is Beta testing?**

- It is testing where real users try the software before it's officially released to find problems in real-life use.

- **What is Component Testing?**

- Component testing are also unit testing where individual modules are tested separately to make sure each module works correctly

- **What is Functional System Testing?**

- Functional System Testing is a type of testing where the complete system is tested to ensure all features and functions work correctly according to the specified requirements.

- **What is Non-Functional Testing?**

- Non-Functional Testing evaluates the quality and performance aspects of a software system, such as speed, security, and usability.

- **What is GUI Testing?**

- GUI (Graphical User Interface) Testing is a type of software testing that verifies whether the visual elements of an application—such as buttons, text boxes, labels, menus, icons, and layouts—work correctly and appear as per the requirements.

- **What is Adhoc testing?**

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

- **What is load testing?**

- Load Testing is a type of performance testing that checks how a software behaves under heavy use or high traffic.

- **What is stress Testing?**

- Stress Testing means checking how much pressure or load a system can handle by pushing it beyond normal limits, to see when and how it fails.

- **What is white box testing and list the types of white box testing?**
 - White Box Testing is a software testing technique in which the internal code structure, logic, and implementation of the application are tested to ensure that the code works correctly.
 - **Types of white box testing:**
 - Unit Testing
 - Statement Coverage
 - Branch Coverage
 - Path Coverage
 - Condition Coverage
 - Loop Testing
- **What is black box testing? What are the different black box testing techniques?**
 - Black Box Testing is a testing method that validates software functionality by checking inputs and expected outputs without examining the internal code.
 - **Black Box Testing Techniques**
 - 1. Equivalence Partitioning**

Divides input data into valid and invalid groups to reduce test cases.
 - 2. Boundary Value Analysis**

Tests values at the edges of input ranges (min, max, just inside/outside).
 - 3. Decision Table Testing**

Uses tables to test different **input combinations and business rules**.
 - 4. State Transition Testing**

Tests system behaviour when moving between different states.
 - 5. Use Case Testing**

Tests end-to-end user scenarios based on use cases.
 - 6. Error Guessing**

Tester predicts possible error-prone areas based on experience.
 - 7. Cause–Effect Graphing**

Identifies logical relationships between inputs (causes) and outputs (effects).

- **Mention what are the categories of defects?**

- 1. Functional Defects**

- The system does not work as expected.
 - Example: Login button not working.

- 2. Performance Defects**

- The system is slow or not responsive.
 - Example: Page takes too long to load.

- 3. Usability Defects**

- The application is hard to use or confusing.
 - Example: Poor layout, unclear error messages.

- 4. Compatibility Defects**

- The application does not work on all devices, browsers, or OS.
 - Example: Works in Chrome but not in Firefox.

- 5. Security Defects**

- The system allows unauthorized access or data leakage.
 - Example: Weak password validation.

- 6. UI (User Interface) Defects**

- Visual issues in the application.
 - Example: Misaligned buttons, wrong font size.

- 7. Data Defects**

- Incorrect data storage or retrieval.
 - Example: Wrong calculation results.

- 8. Documentation Defects**

- Errors in manuals or help documents.
 - Example: Incorrect steps mentioned.

- **Mention what Big bang testing is?**

- Big Bang Testing is an integration testing approach in which all modules of a software application are combined at once and tested as a complete system, rather than integrating and testing modules one by one.

- **What is the purpose of exit criteria?**

- The purpose of exit criteria is to define the conditions under which testing can be stopped and the software can be considered ready for release.

- **When should "Regression Testing" be performed?**
 - Regression Testing should be performed whenever changes are made to the software to ensure that existing functionality still works correctly.
- **What is 7 key principles? Explain in detail?**
 - 1. Testing shows the presence of defects**
 - Testing can find bugs but cannot prove that software is completely error-free.
 - 2. Exhaustive testing is impossible**
 - Testing all possible inputs and combinations is not feasible, so testing must be prioritized.
 - 3. Early testing saves time and cost**
 - Starting testing early in the development life cycle reduces defect-fixing cost and effort.
 - 4. Defect clustering**
 - A small number of modules usually contain most of the defects, so focus should be on high-risk areas.
 - 5. Pesticide paradox**
 - Repeating the same test cases will not find new defects; test cases must be regularly updated.
 - 6. Testing is context dependent**
 - Testing approaches vary based on the type of application and project requirements.
 - 7. Absence-of-errors fallacy**
 - Software without defects is useless if it does not meet user needs and expectations.
- **Difference between QA v/s QC v/s Tester**

| Testing | QC | QA |
|--|------------------------|-------------------------|
| Subset of QA | Quality control | Quantity assurance |
| Focus on test execution | Focus on product | Focus on process |
| Actual testing | Finds defect | Prevents defect |
| Done during development or after development | Done after development | Done before development |

- **Difference between Smoke and Sanity?**

| Smoke | Sanity |
|--|--|
| To check whether build is stable or not | To check specific function is working or not |
| When we get a new build | When any new feature is added or bug fix |
| Basic and critical functionality | Bug fix or related feature |
| Explore: - app install, login, dashboard, logout, app crash or not | Scenario: - app crashes with invalid data login feature (login, dashboard, logout) |

- **Difference between verification and Validation**

| Verification | Validation |
|---|---|
| Checks whether the product is built according to requirements | Checks whether the product meets user needs |
| “Are we building the product, right?” | “Are we building the right product?” |
| Static testing (review of document) | Dynamic testing (live testing of actual software) |
| Performed during development | Performed after development |
| Ex. reviews, inspections, walkthroughs | Ex. testing activities |

- **Difference between Severity and Priority**

| Severity | Priority |
|---|--|
| Severity indicates the impact of a defect on the system | Priority indicates the urgency to fix the defect |
| Focus on business importance | Focus on technical impact |
| Example Minor UI issue needed urgently | Example Application crash |

- **Explain types of Performance testing.**

- The main types of performance testing are:

- 1. Load Testing**

- Tests system behaviour under expected normal user load to ensure the application performs well under normal conditions

- 2. Stress Testing**

- Tests the system beyond its capacity to find the breaking point and observe how it recovers after failure.

- 3. Spike Testing**

- Evaluates performance when there is a sudden increase or decrease in users/traffic.

- 4. Endurance (Soak) Testing**

- Runs the system under normal load for a long time to detect memory leaks or performance degradation.

- 5. Volume Testing**

- Checks system behavior when handling a large amount of data in the database.

- 6. Scalability Testing**

- Determines maximum user load the software application can handle.

- **What is Bug Life Cycle?**

- Bug Life Cycle is the sequence of states a defect goes through from the time it is identified until it is fixed and closed.

Typical Bug Life Cycle Stages:

- 1. New** – Tester finds and reports a bug.
- 2. Assigned** – Bug is assigned to a developer.
- 3. Open** – Developer starts working on the bug.
- 4. Fixed** – Bug is fixed by the developer.
- 5. Retest** – Tester retests the bug
- 6. Verified** – Tester confirms the bug is fixed.
- 7. Closed** – Bug is closed.

Other Possible States:

- **Reopen** – Bug appears again after fix.
- **Deferred** – Bug is not valid.
- **Rejected** – Bug is postponed for later release.
- **Duplicate** – Bug already reported.

- **Explain the difference between Functional testing and Nonfunctional testing**

| Functional testing | Non-functional testing |
|---|---|
| Functional testing is executed first | Non-functional testing should be performed after functional testing |
| Describes what the product does | Describes how good the product works |
| Easy to do manual testing | Tough to do manual testing |
| Types: - Unit testing Smoke testing Sanity testing Black box testing White box testing | Types: - Performance testing Load testing Volume testing Stress testing Security testing |

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

| STLC (Software Testing Life Cycle) | SDLC (Software Development Life Cycle) |
|---|---|
| STLC is to ensure software quality by identifying defects | SDLC is to develop and deliver software as per requirements |
| STLC is limited to testing activities only | SDLC is covers end-to-end software development |
| Quality-assured and defect-free software | Fully developed and functional software |

- **What is priority?**
 - Priority is the level of importance or urgency assigned to a defect, indicating how soon it should be fixed.
- **What is severity?**
 - Severity is the measure of how much impact a defect has on the system or application functionality.

- **What is the difference between test scenarios, test cases, and test script?**
 - **Test Scenario:** What to test
→ A big idea of testing
Example: “Check login feature”
 - **Test Case:** How to test
→ Step-by-step instructions
Example: “Enter username, enter password, click login, check result”
 - **Test Script:** Code to test automatically
→ Program that runs the test
Example: A script that logs in by itself
- **Explain what Test Plan is?**
 - A test plan is a document that explains how testing will be done for a project.
- **What is the information that should be covered.**
 1. **Test objectives** – What needs to be tested
 2. **Scope of testing** – What is in scope and out of scope
 3. **Test strategy** – How testing will be done
 4. **Test environment** – Hardware and software setup
 5. **Test schedule** – Start and end dates of testing
 6. **Resources** – Testers and their roles
 7. **Test tools** – Tools used for testing
 8. **Entry and Exit criteria** – When to start and stop testing
 9. **Risks and assumptions** – Possible issues and conditions
- **Bug categories are...**
 1. **Functional** – something doesn’t work
 2. **UI** – looks wrong
 3. **Usability** – hard to use
 4. **Performance** – slow
 5. **Security** – not safe
 6. **Compatibility** – works on one device/browser but not another
 7. **Data** – wrong data shown or saved

- **Advantage of Bugzilla.**

- Easy to report bugs
- Easy to track bug status (new, fixed, closed)
- Many people can work together
- Keeps all bug details in one place
- Good for large projects

- **Difference between priority and severity**

| Priority | Severity |
|---|--|
| priority indicates the impact of a defect on the system | Severity indicates the urgency to fix the defect |
| Focus on business importance | Focus on technical impact |
| Example Minor UI issue needed urgently | Example Application crash |

- **What are the different Methodologies in Agile Development Model?**

- 1. Scrum**

- Most widely used Agile framework
- Work is divided into Sprints (2–4 weeks)
- Key roles: Product Owner, Scrum Master, Development Team

- 2. Kanban**

- Visual workflow using a Kanban board
- Focuses on continuous delivery
- Limits work in progress (WIP)

- **Explain the difference between Authorization and Authentication in Web testing.**

| Authentication | Authorization |
|---|---|
| Authentication is the process of verifying who the user is. It checks user credentials such as username, password, OTP, or biometrics. | Authorization is the process of verifying what the authenticated user is allowed to do. It controls access to features, pages, or data based on user roles (admin, user, guest). |
| Authentication verifies who the user is (login). | Authorization verifies what the user can access (permissions). |

- **What is the procedure for gui testing?**

- **Understand the design**

- Look at the design or mockups and know how the screen should look.

- **Check screen elements**

- Make sure buttons, icons, menus, text boxes, and images are visible and placed correctly.

- **Test actions**

- Click buttons, links, and menus to see if they work properly.

- **Check text and labels**

- See if text is readable, spelled correctly, and makes sense.

- **Test different screens**

- Open the app on different screen sizes or browsers (if needed).

- **Check error messages**

- Enter wrong input and see if helpful messages appear.

- **Report problems**

- Note anything that looks wrong or doesn't work.

- **When to use usability testing**

- Before releasing a website or app

- While designing it, to see if it makes sense

- After making changes, to avoid new problems

- When users are confused or make mistakes

- Example:

- If people can't find a button or don't know what to do next, usability testing helps you see that.

- **What are the common problems faced in Web testing?**

- Common problems faced in web testing include Browser issues, Slow loading, UI issues, Broken links, Login issues, Data issues, Security issues, Network issues, Device issues

- **What is Error, Defect, Bug and failure?**

- **Error:** - A mistake in coding is called error

- **Defect:** - Error found by tester is called defect

- **Bug:** - defect accepted by development team then it is called bug

- **failure:** - build does not meet the requirements then it is failure