**Introduction to Software Testing**

**What is Testing: -**

**When we say testing, we think of it as:**

* A process of verifying and validating a software, website or application.
* Ensuring product is bug-free\*\*
* Ensuring the product meets business requirement.
* Ensuring the product meets technical requirement.
* Ensuring the product efficiently handles all use cases.

**Official Definition (Source ISTQB):**

Testing is the process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.

**Simple Definition:**

***It’s a process of checking developed software is working as per customer business requirements or not.***

**Objectives of Testing: -**

* Bug detection.
* Bug prevention.
* Assure quality of software/Product.
* Increase customer satisfaction.
* Security, usability, performance.
* Reduce cost.
* Improvement in process.

**What to test in now-a-days applications (e.g., Web Apps):**

* Functionalities.
* GUI
* Usability
* Load
* Performance
* Security
* Compatibility etc.

**Common terms in testing: -**

**Error:** A mistake, misconception, or misunderstanding made by humans that leads to wrong results or discrepancy. (found by developer during unit testing)

**Bug/ Defect /Fault :** The variation between the actual results and expected results. (found by tester in System testing)

**Failure:** If end user identifies a problem while using the application then it is called Failure.

**Debugging:** The process of finding, analyzing and removing the cause of failure in software.

**Verification:** The process of evaluating work-products (not final product) to determine whether they meet the specified requirements. “*Are we building the product right?*”

* Checking documents, codes, designs
* Non-execution testing (static testing).
* Happens before validation.

**Validation:** The process of evaluating software during or at end of development process to determine whether it satisfied business requirements. *“Are we building the right product*?”.

* Testing actual product.
* Execution testing (dynamic testing).

**Black Box Testing:** Testing without any reference to internal structure of component/system.

**White Box Testing:** Testing based on an analysis of internal structure of component/system.

**Severity:** Indicates the degree of impact the defect has on functionality of component or system (Testers evaluation). Functional importance.

**Priority:** Indicates how quickly the bug should be fixed. (Business evaluation).

* **High Severity & High Priority:** Key features failed and no workaround e.g. Login button not working
* **High Severity & Low priority:** Key features failed but there is no impact on customers’ business. E.g. Calculation fault in yearly report which end user won’t use regularly.
* **Low Severity & High priority:** Basic features failed butthere is huge impact on customer’s business. E.g. Misspelled company Logo.
* **Low Severity & Low priority:** Cosmetic issues. E.g. Font family mismatch in a report.

**Risk:** Probability of an event, hazard, accident and its impact (when happens).

**Risk based Testing:** Testing oriented and prioritized on the basis of importance of a feature or component, and its likely chance of failure.

**Static Testing:** Testing of a component or system without execution of that software.

**Dynamic Testing:** Testing that involves execution of the software of a component or system.

**Agile Testing:** Testing practices for a project using agile methodologies, such as emphasizing the test-first design (TDD).

**Continuous Testing:** Testing practice that involves the process of testing early on, testing more often, testing everywhere, and automating.

**STLC: Software Testing Life Cycle**

Testing process with specific steps to be executed in a pre-defined sequence to ensure that quality goals are met.

* Consists of 6 phases.
* Pre-defined sequence.
* All phases are necessary to achieve desired quality.
* Subset of software development life cycle (SDLC).
* Used by testing team.

**Phases of STLC: -**

Requirement Analysis

↓

Test Planning

↓

Test design(Development)

↓

Environment Setup

↓

Test Execution & reporting

↓

Test closure

**Software Testing Models: -**

1. **Waterfall Model: -**

Most basic, highly used but now replacing with agile.

1. **V-Model: -**

Superior than waterfall, but rigid model.

1. **Rapid Application Development (RAD) Model: -**

Incremental, module based, costly & complex.

In this component are development in parallel & finally integrated.

1. **Spiral Model: -**

For complex system, focus on risk analysis

For large & complex projects only.

1. **Agile Model: -**

Most common used now-a-days

Ensure continuous development in line with changing business requirements.

**Principles of Testing: -**

There are 7 principle of testing:

* Testing shows presence of defects.
* Exhaustive testing is not possible.
* Early testing.
* Defect clustering.
* Pesticides paradox.
* Testing is context dependent.
* Absence of errors fallacy.

**Static Testing :**

Static testing is a software testing technique in which the software is tested without the entire program or the software is being run or executed. Static Testing is done in the developmental stage itself so that the errors and bugs can be reported earlier and the dynamic testing test can be carried out then.

The static testing is carried out in organization by conducting reviews and walkthroughs.

We list below on which the static testing is going to be performed:

* Business Requirements Documents.
* A prototype of the software.
* User Manual or training guides or any related documentation
* Performance test script
* Test Cases
* Functional Requirements
* Traceability matrix documents
* Test plan
* Test Data
* Unit Test Cases.

**Types of reviews:**

1. **Requirement Reviews:**

These reviews are carried out by domain experts to find mistakes at requirements

1. **Design Reviews:**

These reviews are carried out by technical experts preferably System or Solution Architects to find mistakes at designs.

1. **Code Reviews:**

These reviews are carried out by developers to find deviation’s in coding standards.

1. **Test Case Reviews:**

These reviews are carried out by senior testers preferably Test Lead to make sure prepared test cases are enough to test the project or module or not.

1. **Formal Reviews:**

If any review activity carried out by following proper review procedures and documentation.

Inspection and Audit are best examples of Formal Reviews.

1. **Inspection:**

If a formal review carried out while executing a task then it is called Inspection.

1. **Audit:**

If a formal review carried out after completion of a task then it is called Audit.

1. **Informal Reviews:**

If any review activity carried out without following proper review procedures and documentation.

1. **Peer Reviews:**

Peer reviews are best examples for informal reviews.

Reviews conducted among the team members are called peer reviews.

**Validation or Dynamic Testing:**

**Unit testing:**

A smallest part in source code of application such as programs or methods / functions etc

is called units

Testing conducted on this units is called unit testing

**Integration Testing :**

once all units are tested , programs are combined together

Testing conducted on overall output is called Integration testing.

**White Box Testing :**

Testing conducted on source code by developers

**Blackbox Testing :**

**System testing :**

Once Integration testing is done , the source code is converted into exectable format

called Build or exe File or ipa file or apk file and deployment into system.

Testing conducted on System or deployed code is called Blackbox testing

System testing will be conducted in some environment:

Dev Environment or Dev Servers

Test Environment or QA Servers or SIT (System Integration Test Environment)

**UAT : User Acceptance Testing**

the code is deployed in separate servers or machine

and will be tested by client or QA people

**Alpha Testing:**

if testing is carried in company premises

**Beta Testing:**

A final round of testing out at client premises

**Functional System testing types :**

**1) Smoke Testing:**

build: an executable file after compiling the source code

it can be URL or .exe or .apk or. ipa.

It is a kind of quick test or rough test performed on given build or application

to decide whether it is eligible for further detail testing or not

what to test?

* 1. we have to check all major features are available or not
  2. we have to check the given application consistently operable or not

Note: Smoke testing is carried out to make sure whether given application

is testable or not. Not to find defect.

**2) Formal Testing:**

If we test a s/w application by following proper documents or procedures (Test Cases)

In simple words, writing test cases and executing test cases is called Formal Testing .

**3) Ad hoc Testing**

If we you test a s/w application without following proper documents or procedures (No Test Cases)

Note: Ad hoc Testing will be performed after Formal Testing if we have enough time.

**4) Priority Based Testing:**

It is a process of deciding what to test first what to test next, what to test last

then we execute the tests in same order

**5) Retesting:**

It is a process of validating reported defects/bugs are correctly resolved or not on modified build.

**6) Regression Testing:**

It is process of identifying various functionalities in modified build whether there is a change of

getting side effects. Then testing those functionalities.

**7) End to End Testing:**

It is process of checking overall functionalities of the system from beginning to end.

This will be carried out by senior testers preferably test lead to make decision whether to stop

testing or not.

**8) Sanity Testing:**

It is process of validating various functionalities depends on bug found in production.

Basically, Sanity testing is subset of regression testing.

**Special Functional System testing types:**

**1. Bench Mark Testing**

It is a process of comparing our product with an established product in market.

**2. Exploratory Testing**

Exploring the application, understanding the application then testing it.

in two scenarios, we go for exploratory testing

i) requirement documentation is not available

ii) if requirements documentation is not providing sufficient inputs

mostly senior domain knowledge will be involved

**3. Monkey or Gorilla or Zig zag**

wantedly performing uneven operations or zig-zag operations with an intension of

making system failure is called monkey testing.

**Non-Functional Testing types:**

Testing performed to check responsiveness, stability and security etc. aspects are

called Non-Functional testing.

**1. User Interface Testing**

User Interface Testing is a software testing in which the front end of the software is tested, whether it is **well equipped, well colored, well aligned, well formatted and well designed**. Frontend means everything including the interface which is visible, accessible or usable by the users.

* + - Testing the size, position, width, height of the elements.
    - Testing of the error messages that are getting displayed.
    - Testing the different sections of the screen.
    - Testing of the font whether it is readable or not.
    - Testing of the screen in different resolutions with the help of zooming in and zooming out.
    - Testing the alignment of the texts and other elements like icons, buttons, etc. are in proper place or not.
    - Testing the colors of the fonts. • Testing whether the image has good clarity or not.
    - Testing the alignment of the images. • Testing of the spelling.
    - The user must not get frustrated while using the system interface.
    - Testing whether the interface is attractive or not.
    - Testing of the scrollbars according to the size of the page if any.
    - Testing of the disabled fields if any.
    - Testing of the size of the images.
    - Testing of the headings whether it is properly aligned or not.
    - Testing of the color of the hyperlink.
    - Testing UI Elements like button, textbox, text area, check box, radio buttons, drop downs, links etc.

**2. Usability**

Usability testing is a way to see how easy to use something is by testing it with real users.

Usability testing is a non-functional software testing, in which the test is performed to measure the degree of ease of access, usability, comfortability and all the attributes which determine the user's experience of using the software.

**3. Performance Testing**

Responsiveness

Stability

WE check following parameter’s:

* time
* speed
* user data volume

**-> Load Testing**

testing the stability of the system under different loads.

we increase the load gradually and test the s/m

**-> Stress Testing**

Suddenly increasing the load and checking stability

**-> Soak Testing**

**-> Volume Testing (user volume)**

Here we test the user data volume capacity that s/m can hold

**-> Spike Testing**

these testing are done using tools

JMeter, load runner, neo load etc. ...

**4) Security Testing**

it is the process of testing how much your application is secured.

-> Authentication Testing

-> Authorization Testing

-> Cookies Testing

-> SQL Injection Testing

**Authentication Testing:**

User is using valid credentials and accessing the application.

Authorization Testing:

User is having limited access rights

User Privileges (Read/Write Access)

**SQL Injection Testing:**

Hacking / Phishing: Stealing your data

SQL Queries .......................

unwanted queries injection.......

**5) Compatibility Testing**

Compatibility testing is a type of software testing that focuses

on ensuring that a software application

or system functions correctly across different environments, platforms, devices and browsers.

The primary objectives of compatibility testing include:

Platform Compatibility: Testing the software on various operating systems (such as Windows, macOS, Linux)

and versions of those systems to ensure it works as expected on each platform.

Browser Compatibility: Ensuring that web applications or websites perform consistently

across different web browsers (e.g., Chrome, Firefox, Safari, Internet Explorer).

**6) Globalization Testing or Internationalization Testing or I18N Testing**

It is a process of checking end user having a choice to configure different language options

(currency, date and time format)

**7) Localization Testing or L10N Testing**

It's a process of checking default format, currency or language for the product designed for

a specific locality of users.

**8)Recovery Testing:**

It's a process of checking how does our application handling unexpected or unpredictable

situations like power failure, system crash, internet issues etc. ...

**9)Installation Testing:**

It's a process of checking are we able to install the software successfully or not

as per guidelines given in installation document.

**10) Uninstallation Testing:**

It's a process of checking are we able to uninstall the s/w successfully or not

also, the data clean up.

**11) Accessibility Testing:**

Accessibility testing is the practice of making your web and mobile apps usable to as many people as possible. It makes apps accessible to those with disabilities, such as vision impairment, hearing disabilities, and other physical or cognitive conditions.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Module 2: software testing project**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

A Test Plan is a document that describes the test scope, test strategy, objectives, schedule, deliverables and resources required to perform testing for a software product

**Test plan template contents:**

▪ Overview

▪ Scope

▪ Inclusions

▪ Test Environments

▪ Exclusions

▪ Test Strategy

▪ Defect Reporting Procedure

▪ Roles/Responsibilities

▪ Test Schedule

▪ Test Deliverables

▪ Pricing

▪ Entry and Exit Criteria

▪ Suspension and Resumption Criteria

▪ Tools

▪ Risks and Mitigations

▪ Approvals

**Test Design Techniques**

Test design techniques helps to design better cases.

Reduce the number of test cases to be executed.

**Techniques: –**

1. Equivalence Class Partitioning
2. Boundary Value Analysis (BVA)
3. Decision Table based testing.
4. State Transition
5. Error Guessing