Unit- IV
Software Testing Types

- Dr. Shivani Budhkar

Scenario Testing

- **What is a Test Scenario?**
- A **TEST SCENARIO** is defined as any functionality that can be tested. It is also called *Test Condition* or *Test Possibility*. As a tester, you should put yourself in the end user's shoes and figure out the real-world scenarios and use cases of the Application Under Test.

- **Scenario Testing**
- **Scenario Testing** in software testing is a method in which actual scenarios are used for testing the software application instead of test cases. The purpose of scenario testing is to test end to end scenarios for a specific complex problem of the software. Scenarios help in an easier way to test and evaluate end to end complicated problems.

Why create Test Scenarios?

- Test Scenarios are created for the following reasons,
- Creating Test Scenarios ensures complete Test Coverage
- Test Scenarios can be approved by various stakeholders like Business Analyst, Developers, Customers to ensure the Application Under Test is thoroughly tested. It ensures that the software is working for the most common use cases.
- They serve as a quick tool to determine the testing work effort and accordingly create a proposal for the client or organize the workforce.
- They help determine the most important end-to-end transactions or the real use of the software applications.
- For studying the end-to-end functioning of the program, Test Scenario is critical.

When not create Test Scenario?

- Test Scenarios may not be created when -
- The Application Under Test is complicated, unstable and there is a time crunch in the project.
- Projects that follow Agile Methodology like Scrum, Kanban may not create Test Scenarios.
- Test Scenario may not be created for a new bug fix or Regression Testing. In such cases, Test Scenarios must be already heavily documented in the previous test cycles. This is especially true for Maintenance projects.

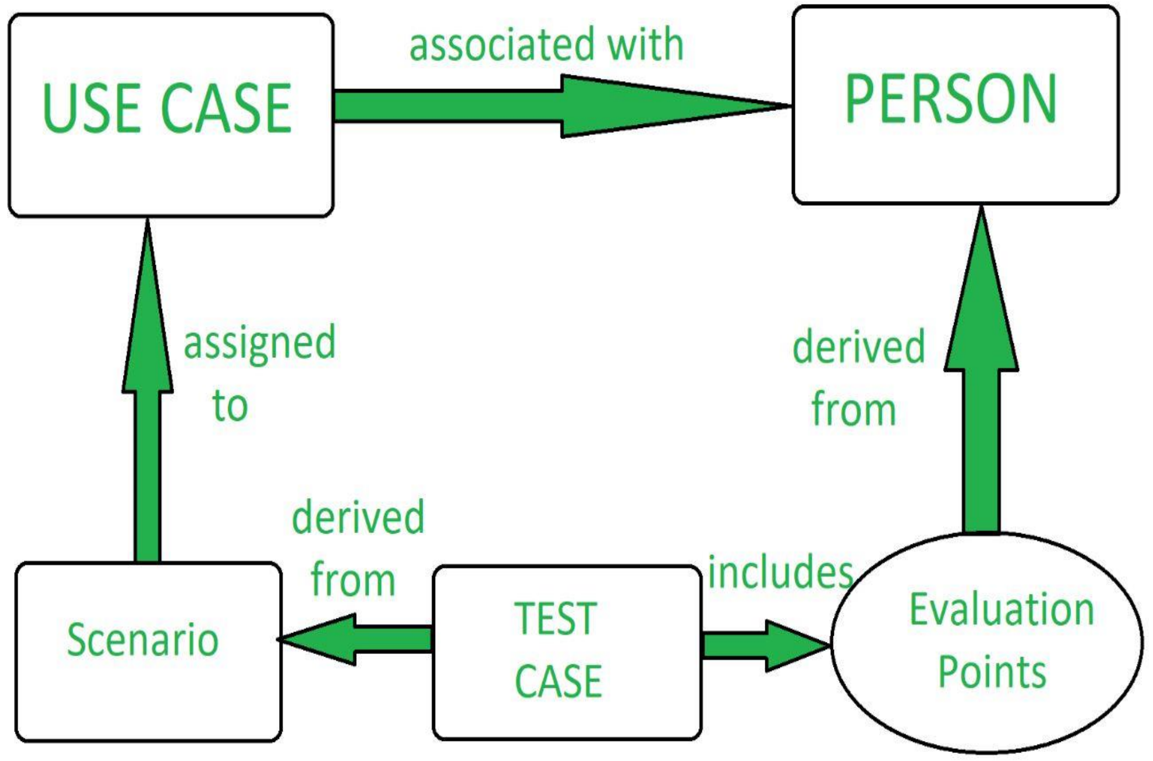
How to Write Test Scenarios

- As a tester, you can follow these five steps to create Test Scenarios-
- **Step 1**: Read the Requirement Documents like BRS, SRS, FRS, of the System Under Test (SUT). You could also refer uses cases, books, manuals, etc. of the application to be tested.
- **Step 2**: For each requirement, figure out possible users actions and objectives. Determine the technical aspects of the requirement. Ascertain possible scenarios of system abuse and evaluate users with hacker's mindset.
- **Step 3:** After reading the Requirements Document and doing your due Analysis, list out different test scenarios that verify each feature of the software.
- **Step 4:** Once you have listed all possible Test Scenarios, a Traceability Matrix is created to verify that each & every requirement has a corresponding Test Scenario
- **Step 5:** The scenarios created are reviewed by your supervisor. Later, they are also reviewed by other Stakeholders in the project.

Tips to Create Test Scenarios

- Each Test Scenario should be tied to a minimum of one Requirement or User Story as per the Project Methodology.
- Before creating a Test Scenario that verifies multiple Requirements at once, ensure you have a Test Scenario that checks that requirement in isolation.
- Avoid creating overly complicated Test Scenarios spanning multiple Requirements.
- The number of scenarios may be large, and it is expensive to run them all. Based on customer priorities only run selected Test Scenarios

Scenario Testing Process:



Performance Testing

- **Performance Testing** is a software testing process used for testing the speed, response time, stability, reliability, scalability and resource usage of a software application under particular workload.
- The main purpose of performance testing is to identify and eliminate the performance bottlenecks in the software application.
- It is a subset of performance engineering and also known as “Perf Testing”.

- The focus of Performance Testing is checking a software program's
  - **Speed** - Determines whether the application responds quickly
  - **Scalability** - Determines maximum user load the software application can handle.
  - **Stability** - Determines if the application is stable under varying loads

Why do Performance Testing?

- Features and Functionality supported by a software system is not the only concern. A software application's performance like its response time, reliability, resource usage and scalability do matter. The goal of Performance Testing is not to find bugs but to eliminate performance bottlenecks.
- Performance Testing is done to provide stakeholders with information about their application regarding speed, stability, and scalability. More importantly, Performance Testing uncovers what needs to be improved before the product goes to market. Without Performance Testing, software is likely to suffer from issues such as: running slow while several users use it simultaneously, inconsistencies across different operating systems and poor usability.
- Performance testing will determine whether their software meets speed, scalability and stability requirements under expected workloads. Applications sent to market with poor performance metrics due to nonexistent or poor performance testing are likely to gain a bad reputation and fail to meet expected sales goals.
- Also, mission-critical applications like space launch programs or life-saving medical equipment should be performance tested to ensure that they run for a long period without deviations.

Types of Performance Testing

- **Load testing -** checks the application's ability to perform under anticipated user loads. The objective is to identify performance bottlenecks before the software application goes live.
- **Stress testing -** involves testing an application under extreme workloads to see how it handles high traffic or data processing. The objective is to identify the breaking point of an application.
- **Endurance testing -** is done to make sure the software can handle the expected load over a long period of time.
- **Spike testing -** tests the software's reaction to sudden large spikes in the load generated by users.
- **Volume testing** - Under Volume Testing large no. of. Data is populated in a database and the overall software system's behavior is monitored. The objective is to check software application's performance under varying database volumes.
- **Scalability testing** - The objective of scalability testing is to determine the software application's effectiveness in "scaling up" to support an increase in user load. It helps plan capacity addition to your software system.

Load Testing

- **Load Testing** is a non-functional software testing process in which the performance of software application is tested under a specific expected load.
- It determines how the software application behaves while being accessed by multiple users simultaneously.
- The goal of Load Testing is to improve performance bottlenecks and to ensure stability and smooth functioning of software application before deployment.
- This testing usually identifies -
- The maximum operating capacity of an application
- Determine whether the current infrastructure is sufficient to run the application
- Sustainability of application with respect to peak user load
- Number of concurrent users that an application can support, and scalability to allow more users to access it.
- It is a type of non-functional testing. In Software Engineering, Load testing is commonly used for the Client/Server, Web-based applications - both Intranet and Internet.

Stress testing

- **Stress Testing** is a type of software testing that verifies stability & reliability of software application.
- The goal of Stress testing is measuring software on its robustness and error handling capabilities under extremely heavy load conditions and ensuring that software doesn't crash under crunch situations.
- It even tests beyond normal operating points and evaluates how software works under extreme conditions.

- Under Stress Testing, AUT is be stressed for a short period of time to know its withstanding capacity.
- A most prominent use **of stress testing is to determine the limit, at which the system or software or hardware breaks**.
- It also checks whether the system demonstrates effective error management under extreme conditions.
- E.g The application under testing will be stressed when 5GB data is copied from the website and pasted in notepad. Notepad is under stress and gives 'Not Responded' error message.

Endurance testing

- **Endurance Testing** is non-functional type of software testing where a software is tested with high load extended over a significant amount of time to evaluate the behaviour of software application under sustained use.
- The main purpose of endurance testing is to ensure that the application is capable enough to handle extended load without any deterioration of response time.
- This type of testing is performed at the last stage of the performance run cycle. Endurance testing is a long process and sometimes lasts for even up to a year. This may include applying external loads such as Internet traffic or user actions.
- This makes endurance testing differ from Load Testing, which usually ends in a couple of hours or so.
- *Endurance means capacity so in other words, you can term Endurance Testing as Capacity Testing.*

Endurance testing

What to monitor in Endurance Testing

- In Endurance Testing following things are tested.
- **Test memory leakage**- Checks are done to verify if there is any memory leakage in the application, which can cause crashing of the system or O.S.
- **Test connection closure between the layer of the system** – If the connection between the layers of the system is not closed successfully, it may stall some or all modules of the system.
- **Test database connection close successfully**- If the database connection is not closed successfully, may result in system crash
- **Test response time** – System is tested for the response time of the system as the application becomes less efficient as a result of the prolonged use of the system.

Spike testing

- **Spike Testing** is a performance testing type used to test software applications with extreme increments and decrements in load.
- The main purpose of spike testing is to evaluate the behaviour of software applications under sudden increment or decrement in user load and determine recovery time after a spike of user load.
- It is performed to estimate weaknesses of software applications.
- The goal of Spike testing is to see how the system responds to unexpected rise and fall of the user load. In Software Engineering Spike testing helps determine system performance will deterioration when there is a sudden high load.
- Another goal of Spike Testing is to determine the recovery time. Between two successive spikes of user load, the system needs some time to stabilize. This recovery time should be as low as possible.

- **Example Spike Testing Scenarios:**
- When an ecommerce store is launching special deals with great discounts such as on Diwali.
- When a web application is live streaming a favourite TV program.

Volume testing

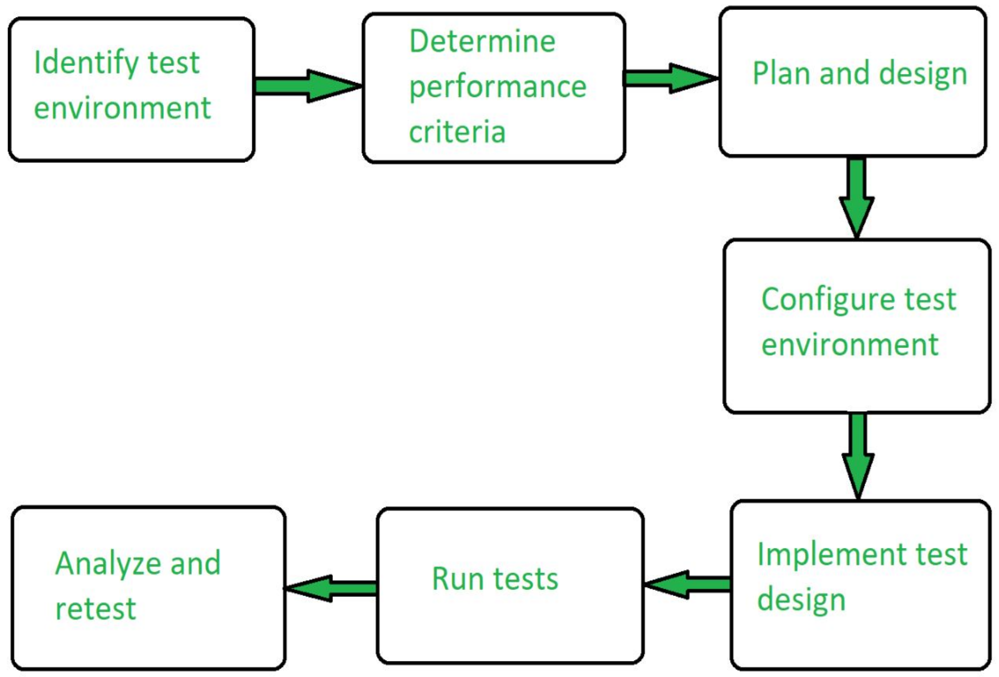
- **VOLUME TESTING** is a type of Software Testing, where the software is subjected to a huge volume of data. It is also referred to as **flood testing.** Volume testing is done to analyse the system performance by increasing the volume of data in the database.
- With the help of Volume testing, the impact on response time and system behaviour can be studied when exposed to a high volume of data.
- For example, testing the music site behaviour when there are millions of user to download the song.
- **Benefits of Volume Testing**
- By identifying load issues, a lot of money can be saved which otherwise will be spent on application maintenance.
- It helps in a quicker start for scalability plans
- Early identification of bottlenecks
- It assures your system is now capable of real-world usage

Scalability testing

- **Scalability Testing** is a non functional testing method that measures performance of a system or network when the number of user requests are scaled up or down.
- The purpose of Scalability testing is to ensure that the system can handle projected increase in user traffic, data volume, transaction counts frequency, etc. It tests system ability to meet the growing needs.
- It is also referred to as performance testing, as such, it is focused on the behaviour of the application when deployed to a larger system or tested under excess load.
- In Software Engineering, Scalability Testing is to measure at what point the application stops scaling and identify the reason behind it.
- **Why do Scalability Testing**
- Scalability testing lets you determine how your application scales with increasing workload.
- Determine the user limit for the Web application.
- Determine client-side degradation and end user experience under load.
- Determine server-side robustness and degradation.

Performance Testing Process

- The methodology adopted for performance testing can vary widely but the objective for performance tests remain the same. It can help demonstrate that your software system meets certain pre-defined performance criteria. Or it can help compare the performance of two software systems. It can also help identify parts of your software system which degrade its performance.
- Below is a generic process on how to perform performance testing


Identify test environment
Determine performance criteria
Plan and design
Configure test environment
Implement test design
Run tests
Analyze and retest

Performance Testing Process

- **Identify your testing environment -** Know your physical test environment, production environment and what testing tools are available. Understand details of the hardware, software and network configurations used during testing before you begin the testing process. It will help testers create more efficient tests. It will also help identify possible challenges that testers may encounter during the performance testing procedures.
- **Identify the performance acceptance criteria -** This includes goals and constraints for throughput, response times and resource allocation. It is also necessary to identify project success criteria outside of these goals and constraints. Testers should be empowered to set performance criteria and goals because often the project specifications will not include a wide enough variety of performance benchmarks. Sometimes there may be none at all. When possible finding a similar application to compare to is a good way to set performance goals.
- **Plan & design performance tests -** Determine how usage is likely to vary amongst end users and identify key scenarios to test for all possible use cases. It is necessary to simulate a variety of end users, plan performance test data and outline what metrics will be gathered.
- **Configuring the test environment -** Prepare the testing environment before execution. Also, arrange tools and other resources.
- **Implement test design -** Create the performance tests according to your test design.
- **Run the tests -** Execute and monitor the tests.
- **Analyse, tune and retest** - Consolidate, analyse and share test results. Then fine tune and test again to see if there is an improvement or decrease in performance. Since improvements generally grow smaller with each retest, stop when bottlenecking is caused by the CPU. Then you may have the consider option of increasing CPU power.

Performance Test Tools

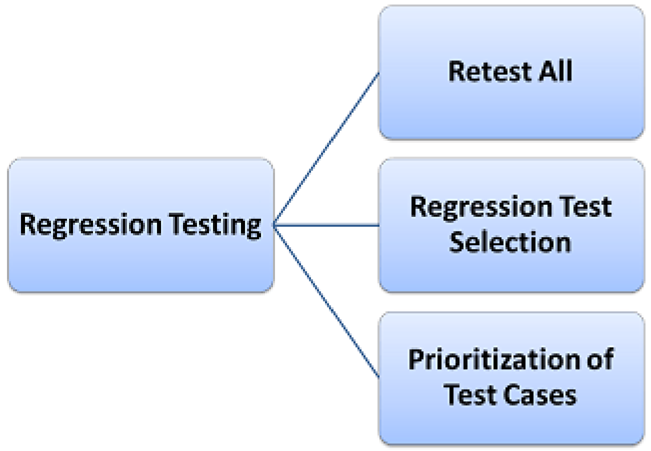
- There are a wide variety of performance testing tools available in the market. The tool you choose for testing will depend on many factors such as types of the protocol supported, license cost, hardware requirements, platform support etc. Below is a list of popularly used testing tools.
- **LoadNinja** – is revolutionizing the way we load test. This cloud-based load testing tool empowers teams to record & instantly playback comprehensive load tests, without complex dynamic correlation & run these load tests in real browsers at scale. Teams are able to increase test coverage. & cut load testing time by over 60%.
- **NeoLoad -** is the performance testing platform designed for DevOps that seamlessly integrates into your existing Continuous Delivery pipeline. With NeoLoad, teams test 10x faster than with traditional tools to meet the new level of requirements across the full Agile software development lifecycle - from component to full system-wide load tests.
- **HP LoadRunner -** is the most popular performance testing tools on the market today. This tool is capable of simulating hundreds of thousands of users, putting applications under real-life loads to determine their behavior under expected loads. Loadrunner features a virtual user generator which simulates the actions of live human users.
- **Jmeter -** one of the leading tools used for load testing of web and application servers.

Regression testing

- **REGRESSION TESTING** is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features.
- Regression Testing is nothing but a full or partial selection of already executed test cases which are re-executed to ensure existing functionalities work fine.
- This testing is done to make sure that new code changes should not have side effects on the existing functionalities. It ensures that the old code still works once the latest code changes are done.
- **Need of Regression Testing**
- The **Need of Regression Testing** mainly arises whenever there is requirement to change the code and we need to test whether the modified code affects the other part of software application or not. Moreover, regression testing is needed, when a new feature is added to the software application and for defect fixing as well as performance issue fixing.

How to do Regression Testing

- In order **to do Regression Testing** process, we need to first debug the code to identify the bugs. Once the bugs are identified, required changes are made to fix it, then the regression testing is done by selecting relevant test cases from the test suite that covers both modified and affected parts of the code.
- Software maintenance is an activity which includes enhancements, error corrections, optimization and deletion of existing features. These modifications may cause the system to work incorrectly. Therefore, Regression Testing becomes necessary. Regression Testing can be carried out using the following techniques:



How to do Regression Testing

- **Retest All**

- This is one of the methods for Regression Testing in which all the tests in the existing test bucket or suite should be re-executed. This is very expensive as it requires huge time and resources.

- **Regression Test Selection**

- **Regression Test Selection** is a technique in which some selected test cases from test suite are executed to test whether the modified code affects the software application or not. Test cases are categorized into two parts, reusable test cases which can be used in further regression cycles and obsolete test cases which can not be used in succeeding cycles.

- **Prioritization of Test Cases**

- Prioritize the test cases depending on business impact, critical & frequently used functionalities. Selection of test cases based on priority will greatly reduce the regression test suite.

Tools for regression testing

- In regression testing, we generally select the test cases form the existing test suite itself and hence, we need not to compute their expected output and it can be easily automated due to this reason. Automating the process of regression testing will be very much effective and time saving.
- Most commonly used tools for regression testing are:
  - Selenium
  - WATIR (Web Application Testing In Ruby)
  - QTP (Quick Test Professional)
  - RFT (Rational Functional Tester)
  - Winrunner
  - Silktest

Types of Regression testing

Various types of regression testing can be taken up to ensure existing functionality is not affected by the recent changes in the application.

- 1. **Corrective** Regression Testing- This type of testing is used when there are no changes introduced in the product's specification. Moreover, the already existing test cases can be easily reused to conduct the desired test.
  2. **Retest-all** Regression Testing - This type of testing is very tedious and tends to waste a lot of time.
- The strategy involves the testing of all aspects of a particular product as well as reusing all test cases even where the changes/modifications have not been made.
- This type of testing is not at all advisable when there is a small change, that has been introduced in the existing product.
- 3. **Selective** Regression Testing -It is done to analyse the impact of new code added to the already existing code of the software.
- When this type of regression testing is conducted, a subset from the existing test cases is used, to reduce the effort required for retesting and the cost involved.
- For example, a test unit is re-run in case there is some change incorporated in the program entities such as functions and variables.

Types of Regression testing

4. **Progressive** Regression Testing - This type of regression testing works effectively when there are certain changes done in the program specifications as well as new test cases are designed.

- Conducting this testing helps in ensuring that, there are no features that exist in the previous version that has been compromised in the new and updated version.

5. **Complete** Regression Testing -Complete regression testing is the best to be used in case there are multiple changes that have been done to the already existing code.

- Moreover, this type of testing is specifically used when the new change has a certain impact on the root code of the software.
- Conducting this type of testing is highly beneficial to identify unexpected issues.
- Once this testing is completed, the final system can be made available to the user.

6. **Partial** Regression Testing -Partial regression testing is done to test issues when new codes are added to already existing code.

- The idea behind partial regression testing to make sure that a system is performing as it is supposed to be after addition of new code.

7. **Unit** Regression Testing -it's the most important part of unit testing. Mostly conducted in isolation, mainly focused on code unit and all the dependencies and interactions are will be blocked at the time of test.

Advantages & Disadvantages of Regression Testing:

- **Advantages of Regression Testing:**
  - It ensures that no new bugs has been introduced after adding new functionalities to the system.
  - As most of the test cases used in Regression Testing are selected from the existing test suite and we already know their expected outputs. Hence, it can be easily automated by the automated tools.
  - It helps to maintain the quality of the source code.
- **Disadvantages of Regression Testing:**
  - It can be time and resource consuming if automated tools are not used.
  - It is required even after very small changes in the code.

Ad hoc Testing

- **Ad hoc Testing** is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage.
- Ad hoc testing is done randomly and it is usually an unplanned activity which does not follow any documentation and test design techniques to create test cases.
- It also known as Random Testing or Monkey Testing
- It is performed without any planning and documentation.
- The tests are conducted informally and randomly without any formal procedure or expected results.
- Ad hoc 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 Software testing technique called **Error Guessing.** Error guessing can be done by the people having enough experience on the system to "guess" the most likely source of errors.
- Ad hoc testing can be performed when there is limited time to do elaborative testing. Usually adhoc testing is performed after the formal test execution. And if time permits, ad hoc testing can be done on the system. Ad hoc testing will be effective only if the tester is knowledgeable of the System Under Test.

Types of Adhoc testing

Buddy Testing	**Two buddies mutually work on identifying defects in the same module. Mostly one buddy will be from development team and another person will be from testing team. Buddy testing helps the testers develop better test cases and development team can also make design changes early. This testing usually happens after Unit Testing completion.**
Pair testing	Two testers are assigned modules, share ideas and work on the same machines to find defects. One person can execute the tests and another person can take notes on the findings. Roles of the persons can be a tester and scriber during testing. Comparison Buddy and Pair Testing: Buddy testing is combination of unit and System Testing together with developers and testers but Pair testing is done only with the testers with different knowledge levels. (Experienced and non-experienced to share their ideas and views)
Monkey Testing	Randomly test the product or application without test cases with a goal to break the system.

Best practices of Adhoc testing

- Following best practices can ensure effective Adhoc Testing.
- **Good business knowledge**

 Testers should have good knowledge of the business and clear understanding of the requirements- Detailed knowledge of the end to end business process will help find defects easily. Experienced testers find more defects as they are better at error guessing.

- **Test Key Modules**

 Key business modules should be identified and targeted for ad-hoc testing. Business critical modules should be tested first to gain confidence on the quality of the system.

- **Record Defects**

 All defects need to be recorded or written in a notepad. Defects must be assigned to developers for fixing. For each valid defect, corresponding test cases must be written & must be added to planned test cases.

 These Defect findings should be made as lesson learned and these should be reflected in our next system while we are planning for test cases.

Usability Testing

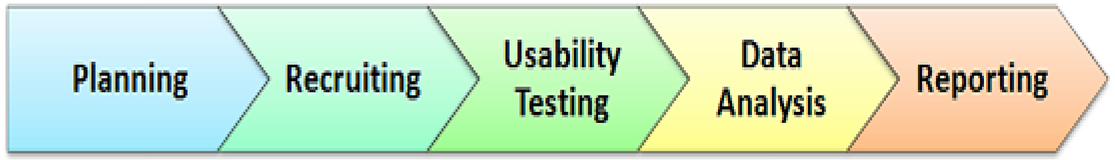
- **Usability Testing** also known as User Experience(UX) Testing, is a testing method for measuring how easy and user-friendly a software application is. A small set of target end-users, use software application to expose usability defects. Usability testing mainly focuses on user's ease of using application, flexibility of application to handle controls and ability of application to meet its objectives.
- This testing is recommended during the initial design phase of SDLC, which gives more visibility on the expectations of the users.
- Aesthetics and design are important. How well a product looks usually determines how well it works.
- Software Engineering, Usability Testing identifies usability errors in the system early in the development cycle and can save a product from failure.

Usability Testing

- The goal of this testing is to satisfy users and it mainly concentrates on the following parameters of a system:
- **The effectiveness of the system**
- Is the system is easy to learn?
- Is the system useful and adds value to the target audience?
- Are Content, Colour, Icons, Images used are aesthetically pleasing?
- **Efficiency**
- Little navigation should be required to reach the desired screen or webpage, and scrollbars should be used infrequently.
- Uniformity in the **format** of screen/pages in your application/website.
- Option to search within your software application or website.
- **Accuracy**
- No outdated or incorrect data like contact information/address should be present.
- No broken links should be present.
- **User Friendliness**
- Controls used should be self-explanatory and must not require training to operate
- Help should be provided for the users to understand the application/website
- Alignment with the above goals helps in effective usability testing

How to do Usability Testing: Complete Process

- **Planning**:- During this phase the goals of usability test are determined. Having volunteers sit in front of your application and recording their actions is not a goal. You need to determine critical functionalities and objectives of the system. You need to assign tasks to your testers, which exercise these critical functionalities. During this phase, the usability testing method, number & demographics of usability testers, test report formats are also determined
- **Recruiting**: During this phase, you recruit the desired number of testers as per your usability test plan. Finding testers who match your demographic (age, sex etc.) and professional (education, job etc.) profile can take time.
- **Usability Testing**: During this phase, usability tests are actually executed.
- **Data Analysis**: Data from usability tests is thoroughly analyzed to derive meaningful inferences and give actionable recommendations to improve the overall usability of your product.
- **Reporting**: Findings of the usability test is shared with all concerned stakeholders which can include designer, developer, client, and CEO



Usability Testing Advantages & Disadvantages

- **Usability Testing Advantages**

- As with anything in life, usability testing has its merits and de-merits. Let's look at them
- It helps uncover usability issues before the product is marketed.
- It helps improve end-user satisfaction
- It makes your system highly effective and efficient
- It helps gather true feedback from your target audience who actually use your system during a usability test. You do not need to rely on "opinions" from random people.

- **Usability Testing Disadvantages**

- Cost is a major consideration in usability testing. It takes lots of resources to set up a Usability Test Lab. Recruiting and management of usability testers can also be expensive

Accessibility Testing

- Accessibility Testing is defined as a type of Software Testing performed to ensure that the application being tested is usable by people with disabilities like hearing, color blindness, old age and other disadvantaged groups. It is a subset of Usability Testing.
- People with disabilities use assistive technology which helps them in operating a software product. Examples of such software are:
- **Speech Recognition Software -** It will convert the spoken word to text , which serves as input to the computer.
- **Screen reader software** - Used to read out the text that is displayed on the screen
- **Screen Magnification Software**- Used to enlarge the monitor and make reading easy for vision-impaired users.
- **Special keyboard** made for the users for easy typing who have motor control difficulties

Why Accessibility Testing?

Reason 1: Cater to market for Disabled People.

- About 20% of the population has disability issues.
- 1 in 10 people have a several disability
- 1 in 2 people over 65 have reduced capabilities
- Disabilities include blindness, deaf, handicapped, or any disorders in the body.
- A software product can cater to this big market, if it's made disabled friendly. Accessibility issues in software can be resolved if Accessibility Testing is made part of normal software testing life cycle.

- **Reason 2**: Abide by Accessibility Legislations

- Government agencies all over the world have come out with legalizations, which requires that IT products to be accessible by disabled people.
- Following are the legal acts by various governments -
- United States: Americans with Disabilities Act - 1990
- United Kingdom: Disability Discrimination Act - 1995
- Australia: Disability Discrimination Act - 1992
- Ireland : Disability Act of 2005
- Accessibility Testing is important to ensure legal compliance.

Why Accessibility Testing?

Reason 3: Avoid Potential Law Suits

- In the past, Fortune 500 companies have been sued because their products were not disabled friendly. Here a few prominent cases
- National Federation for the Blind (NFB) vs Amazon (2007)
- Sexton and NFB vs Target (2007)
- NFB Vs AOL settlement (1999)
- It's best to create products which support disabled and avoid potential lawsuits.

Which Disabilities to Support?

- Application must support people with disabilities like -

Type of Disability	Disability Description
Vision Disability	· Complete Blindness or Color Blindness or Poor Vision · Visual problems like visual strobe and flashing effect problems
Physical Disability	· Not able to use the mouse or keyboard with one hand. · Poor motor skills like hand movements and muscle slowness
Cognitive disability	· Learning Difficulties or Poor Memory or not able to understand more complex scenarios
Literacy Disability	· Reading Problems
Hearing Disability	· Auditory problems like deafness and hearing impairments · Cannot able to hear or not able to hear clearly

Accessibility Testing Tools

- To make your website more acceptable and user-friendly, it is crucial that it is easily accessible. There are various tools which can check the accessibility of the website. Some of these popular tools are listed below-
    1) Wave
    2) TAW
    3) Accessibility Valet
    4) Accessibility Developer Tools
    5) Quick Accessibility Page Tester
    6) aDesigner
    7) WebAnywhere
    8) Web accessibility toolbar

GUI Testing

Graphical User-interface Testing or GUI testing is a process of testing the user interface of an application.

A graphical user interface includes all the elements such as menus, checkbox, buttons, colors, fonts, sizes, icons, content, and images. GUI testing is done to check the functionality and usability of design elements as a user for an application under test.

Why do you need GUI testing?

- Modern applications are beyond the desktop they are either mobile based or cloud-based applications. They need to be more user-friendly as per customer demand. The application interface and user experience play a significant role in application success as it is released to the market. A GUI testing team always pays close attention to each detail in visual dynamics to ensure end-user satisfaction and ease.
- It tests the various aspects of the user interface, such as:
- Visual Design
- Functionality
- Security
- Compliance
- Usability
- Performance

GUI Testing

- **Benefits of using GUI testing are:**

 It releases an error-free application software

 It increases the efficiency of software

 Improves software quality

- **What we check in GUI Testing?**

 It extensively checks the user-interface of the application under test.

 Testing the size, position, height, width of the visual elements

 Verifying and testing the error messages are displayed or not

 Testing different sections of the display screen

 Verifying the usability of carousel arrows

 Checking the navigation elements at the top of the page

 Checking the message displayed, frequency and content

 Verifying the functionality of proper filters and ability to retrieve results.

 Checking alignment of radio buttons, drop downs

 Verifying the title of each section and their correctness

 Cross-checking the colors and its synchronization with the theme

GUI Testing Approaches

- **Manual Testing**

 This approach involves human tester, where each screen is manually checked to validate each functionality by creating and executing test cases. It is a useful approach when part of UI or a feature is ready, the probability of defects is more at the initial stage, and human intervention is required.

 It is convenient to use where the UI is unstable and go through a lot of changes. It is viable for quick checks which can be done at any moment. Moreover, manual testing requires expertise and skills to validate design elements which are not possible without a human tester.

- **Record and Replay Testing**

 GUI record and replay tools are used to test applications for their user interface. Using such tools, testers run an application and record the user interaction with the app. A script runs to track and save the user actions, including cursor movements, which can be replayed several times to find the issues in the interface.

 It also supports automated regression testing. It is can be used for cross-browser testing. It is a convenient and lightweight solution for testing. It doesn't work well where you have lots of iterations in the GUI of the applications. Recapturing and replaying test cases to check functionality is time-consuming, and tracking their updated version is a cumbersome process.

GUI Testing Approaches

- **Model-based testing**
  - In this type of **GUI testing**, a model is created to understand and evaluate the system's behavior. This approach is useful in creating accurate test cases using system requirements. It is a structured, thorough, measurable form of testing.
  - There are three essential aspects of model-based GUI testing:
  - Automatically generated test cases from the model
  - Manually derived test cases from the model
  - Model and requirements coverage metrics
  - There are tools which could automatically generate test based on the model.
- **Things to consider for model-based testing:**
  - Create the model
  - Determine the information as inputs in the system
  - Verifying the expected output
  - Execute tests
  - Checking and validating actual vs. expected
  - Take further action on the model

GUI Testing Approaches

- You can even derive test cases for GUI in two ways:
- Charts: The charts display the state of the system and check the state after some input.
- Decision Tables: This helps in deciding the results for each input
- Model-Based testing is preferred as the technique aligns with requirements which define even the undesirable states a GUI can attain.

- Thus, GUI testing is crucial to the successful release of the software as it validates the user experience. GUI testing is helps to deliver high-quality and **user-friendly software**. You achieve a higher level of user engagement and satisfaction.

Tools available to conduct automated UI Test

- AutoHotkey
- Selenium
- Sikuli
- Robot Framework
- Water
- Dojo Toolkit
- QTP
- Cucumber
- SilkTest
- TestComplete
- Squish GUI Tester

Validation testing

- Validation testing is the process of ensuring if the tested and developed software satisfies the client /user needs. The business requirement logic or scenarios have to be tested in detail. All the critical functionalities of an application must be tested here.
- As a tester, it is always important to know how to verify the business logic or scenarios that are given to you. One such method that helps in detail evaluation of the functionalities is the Validation Process.
- Whenever you are asked to perform a validation test, it takes a great responsibility as you need to test all the critical business requirements based on the user needs. There should not be even a single miss on the requirements asked by the user. Hence a keen knowledge on validation testing is much important.
- As a tester, you need to evaluate if the test execution results comply with that mentioned in the requirements document. Any deviation should be reported immediately and that deviation is thus called a bug.
- Tools like HP quality Centre, Selenium, Appium, etc are used to perform validation test and we can store the test results there. A proper test plan, test execution runs, defect reports, reports & metrics are the important deliverables to be submitted.

Validation testing

- **From a company perspective, the validation test in simple is carried out by the following steps:**
- You gather the business requirements for validation testing from the end user.
- Prepare the business plan and send it for the approval to the onsite/stakeholders involved.
- On approval of the plan, you begin to write the necessary test cases and send them for approval.
- Once approved you begin to complete testing with the required software, environment and send the deliverables as requested by the client.
- Upon approval of the deliverables, UAT testing is done by the client.
- After that, the software goes for production.

Specification-based testing

Specification Based Testing Technique is also known as ***Behavior Based Testing*** and ***Black Box Testing*** techniques because in this testers view the software as a black-box.

- *Both **<u>Functional Testing</u>** and **Non-Functional Testing** is a type of Specification Based Testing.*
- ***Specification Based Test Design Technique*** uses the specification of the program as the point of reference for test data selection and adequacy. A specification can be anything like a written document, collection of use cases, a set of models or a prototype.

Types of Specification Based Testing Techniques

- ***Equivalence Partitioning:*** ***Software Testing*** *technique that divides the input data of a software unit into partitions of equivalent data from which test cases can be derived.*
- ***Boundary Value Analysis:*** ***Software Testing*** *technique in which tests are designed to include representatives of boundary values in a range.*
- ***Decision Tables:*** ***Software Testing*** *technique in which tests are more focused on business logic or business rules. A decision table is a good way to deal with combinations of inputs.*
- ***State Transitioning:*** ***Software Testing*** *technique which is used when the system is defined in terms of a finite number of states and the transitions between the states is governed by the rules of the system.*