**Question 1: Write a manual test case.**

Ans: Manual testing is a type of software testing where testers evaluate software or application quality manually, without the help of automated testing tools or executing test scripts. Testers interact with the system similar to how an end user would to identify bugs, defects, and issues in the software that create friction in user experience.

When a developer manually runs their application and tries out the features they have coded, they are performing manual testing. Its simplicity makes manual testing great for small-scale testing of personal projects. Even for large-scale testing where there are thousands and millions of items and features to be tested, manual testing is still needed to check for automation feasibility.

I can say,” Manual testing is a good old-fashioned human instinct and attention to detail.” Manual testing involves testing software manually, i.e., without using any automated tools. For executing this testing, you sometimes create test cases beforehand. These test cases are called manual test cases. A few examples of manual test cases include:

* validating that the login page is working properly,
* testing that the correct data is displayed on the search results page,
* ensuring the registration form is accurate before submission,
* testing the functioning of the shopping cart, and
* validating the order process.

**How to write test cases:**

**Step 1 – Test Case ID:**

In this step, the tester will assign a unique identifier to the test case. This allows the tester to recall and identify the test case in the future easily.

**Example:** TC-01: Verify Login Functionality for a User

**Step 2 – Test Case Description:**

The tester will describe the test case, outlining what it is designed to do. The tester may also provide a brief overview of the expected behavior. **An Example:**Test Case Description: Test for Logging Into the application Given: A valid username and password for the web application When: User enters the username and password in the login page Then: the user should be able to log in to the application successfully. The Home page for the application should be displayed.

**Step 3 – Pre-Conditions:**

The tester will document any pre-conditions that need to be in place for the test case to run properly. It may include initial configuration settings or manually executing some previous tests. A Pre-Condition example in testing could be that the test environment must be set up, to be very similar to the production environment, including the same hardware, operating system, and software.

**Step 4 – Test Steps:**

The tester will document the detailed steps necessary to execute the test case. This includes deciding which actions should be taken to perform the test and also possible data inputs.

**Example steps for our login test:**

1. Launch the login application under test.

2. Enter a valid username and password in the appropriate fields.

3. Click the ‘Login’ button.

4. Verify that the user has been successfully logged in.

5. Log out and check if the user is logged out of the system.

**Step 5 – Test Data:**

The tester will define any necessary test data. For example, if the test case needs to test that login fails for incorrect credentials, then test data would be a set of incorrect usernames/passwords.

**Step 6 – Expected Result:**

The tester will provide the expected result of the test. This is the result the tester is looking to verify. **Examples of how to define expected results:**

1. A user should be able to enter a valid username and password and click the login button.

2. The application should authenticate the user’s credentials and grant access to the application.

3. The invalid user should not be able to enter the valid username and password; click the login button.

4. The application should reject the user’s credentials and display an appropriate error message.

**Step 7 – Post Condition:**

The tester will provide any cleanup that needs to be done after running the test case. This includes reverting settings or cleaning up files created during the test case. **Example:** 1. The user can successfully log in after providing valid credentials. 2. After providing invalid credentials, The user is shown the appropriate error message. 3. The user’s credentials are securely stored for future logins. 4. The user is taken to the correct page after successful login. 5. The user cannot access the page without logging in. 6. No unauthorized access to the user’s data.

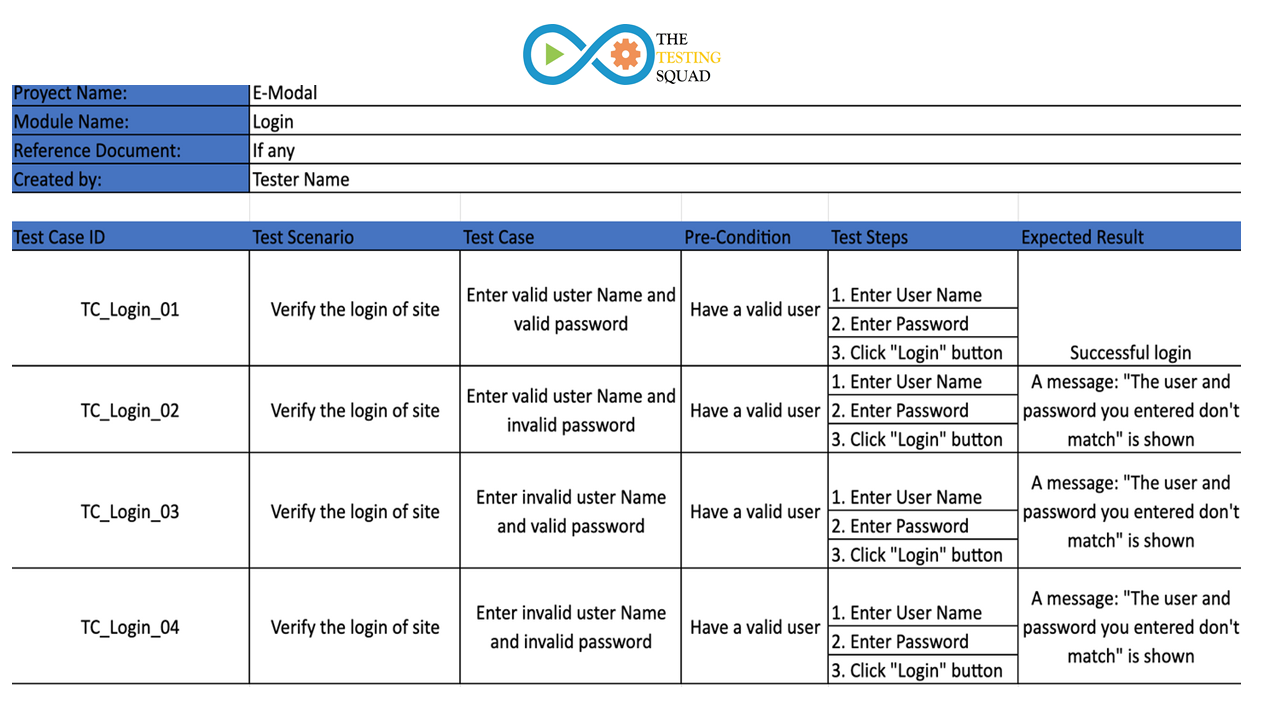
**Step 8 – Actual Result:**

The tester will document the actual result of the test. This is the result the tester observed when running the test. **Example**: After entering the correct username and password, the user is successfully logged in and is presented with the welcome page.

**Step 9 – Status:**

The tester will report the status of the test. If the expected and actual results match, the test is said to have passed. If they do not match, the test is said to have failed.

**Example**: Tested the valid login functionality. Result: The user is able to log in with valid credentials. Overall Test Result: All the test steps were successfully executed, and the expected results were achieved. The login application is functioning as expected. Tested for Invalid Login functionality. Result: The user is unable to log in with invalid credentials. Overall Test Result: The invalid login functionality has been tested and verified to be working as expected



**Question-2:**

**1. Unit Testing:** Unit Testing is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

In SDLC, STLC, V Model, Unit testing is first level of testing done before integration testing. Unit testing is a WhiteBox testing technique that is usually performed by the developer. Though, in a practical world due to time crunch or reluctance of developers to tests, QA engineers also do unit testing.

Unit Testing is important because software developers sometimes try saving time doing minimal unit testing and this is myth because inappropriate unit testing leads to high cost[Defect](https://www.guru99.com/defect-management-process.html)fixing during [System Testing](https://www.guru99.com/system-testing.html), [Integration Testing](https://www.guru99.com/integration-testing.html) and even Beta Testing after application is built. If proper unit testing is done in early development, then it saves time and money in the end.

Here, are the key reasons to perform unit testing in software engineering:

1. Unit tests help to fix bugs early in the development cycle and save costs.
2. It helps the developers to understand the testing code base and enables them to make changes quickly
3. Good unit tests serve as project documentation
4. Unit tests help with code re-use. Migrate both your code **and** your tests to your new project. Tweak the code until the tests run again.

**Unit Testing Techniques:**

The **Unit Testing Techniques** are mainly categorized into three parts which are Black box testing that involves testing of user interface along with input and output, White box testing that involves testing the functional behaviour of the software application and Gray box testing that is used to execute test suites, test methods, test cases and performing risk analysis.

Code coverage techniques used in Unit Testing are listed below:

* Statement Coverage
* Decision Coverage
* Branch Coverage
* Condition Coverage
* Finite State Machine Coverage

**Unit Testing Example: Mock Objects:**

Unit testing relies on mock objects being created to test sections of code that are not yet part of a complete application. Mock objects fill in for the missing parts of the program.

For example, you might have a function that needs variables or objects that are not created yet. In unit testing, those will be accounted for in the form of mock objects created solely for the purpose of the unit testing done on that section of code.

**Unit Testing Tools:**

There are several automated unit test software available to assist with unit testing in software testing. We will provide a few examples below:

1. [Junit](https://www.guru99.com/junit-tutorial.html): Junit is a free to use testing tool used for Java programming language. It provides assertions to identify test method. This tool test data first and then inserted in the piece of code.
2. [NUnit](https://nunit.org/): NUnit is widely used unit-testing framework use for all .net languages. It is an open source tool which allows writing scripts manually. It supports data-driven tests which can run in parallel.
3. [JMockit](http://jmockit.github.io/index.html): JMockit is open source Unit testing tool. It is a code coverage tool with line and path metrics. It allows mocking API with recording and verification syntax. This tool offers Line coverage, Path Coverage, and Data Coverage.
4. [EMMA](http://emma.sourceforge.net/): EMMA is an open-source toolkit for analyzing and reporting code written in Java language. Emma support coverage types like method, line, basic block. It is Java-based so it is without external library dependencies and can access the source code.
5. [PHPUnit](https://phpunit.de/): PHPUnit is a unit testing tool for PHP programmer. It takes small portions of code which is called units and test each of them separately. The tool also allows developers to use pre-define assertion methods to assert that a system behave in a certain manner.

Those are just a few of the available unit testing tools. There are lots more, especially for [C languages](https://www.guru99.com/c-programming-language.html) and Java, but you are sure to find a unit testing tool for your programming needs regardless of the language you use.

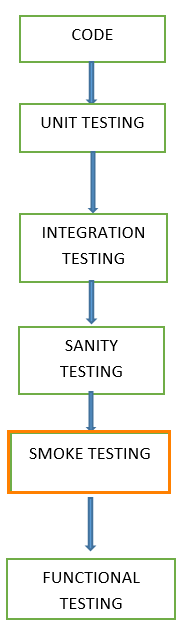
In conclusion, unit testing is a crucial part of the software development process that helps ensure the quality and reliability of your code. By writing and running small, isolated tests for individual code units, you can catch errors and bugs early on, improve the maintainability of your code, and build confidence in your work.

**2. Smoke Testing:**  It is a software testing process that determines whether the deployed software build is stable or not. Smoke testing is a confirmation for QA team to proceed with further software testing. It consists of a minimal set of tests run on each build to test software functionalities. Smoke testing is also known as “Build Verification Testing” or “Confidence Testing.”

In simple terms, smoke tests means verifying the important features are working and there are no showstoppers in the build that is under testing. It is a mini and rapid regression test of major functionality. It is a simple test that shows the product is ready for testing. This helps determine if the build is flawed as to make any further testing a waste of time and resources.

**When do we do smoke testing?**

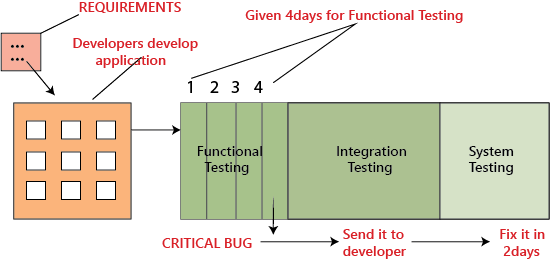
Smoke Testing is done whenever the new functionalities of software are developed and integrated with existing build that is deployed in QA/staging environment. It ensures that all critical functionalities are working correctly or not.



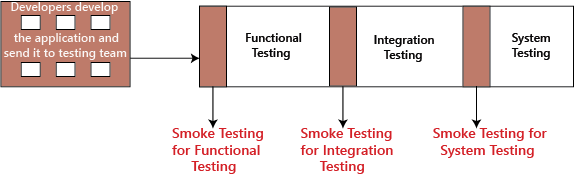
**Scenarios 1**

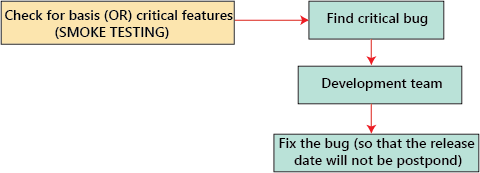
The developer develops the application and handed over to the testing team, and the testing team will start the functional testing

Suppose we assume that four days we are given to the **functional testing**. On the first day, we check one module, and on the second day, we will go for another module. And on the fourth day, we find a **critical bug** when it is given it to the developer; he/she says it will take another two days to fix it. Then we have to postpone the release date for these extra two days.



To overcome this problem, we perform **smoke testing**, let us see how it works, in the above situation, instead of the testing module by module thoroughly and come up with critical bug at the end, it is better to do **smoke testing** before we go for functional, integration and system testing that is, in each module we have to test for essential or critical features, and then proceed for further testing as we can see in the below images:





In Software Engineering, Smoke testing should be performed on each and every build without fail as it helps to find defects in early stages. Smoke test activity is the final step before the software build enters the system stage. Smoke tests must be performed on each build that is turned to testing. This applies to new development and major and minor releases of the system.

**3. User Acceptance:** User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.

**Purpose of UAT:**

The main **Purpose of UAT** is to validate end to end business flow. It does not focus on cosmetic errors, spelling mistakes or system testing. User Acceptance Testing is carried out in a separate testing environment with production-like data setup. It is kind of black box testing where two or more end-users will be involved.

UAT is performed by –

* Client
* End users

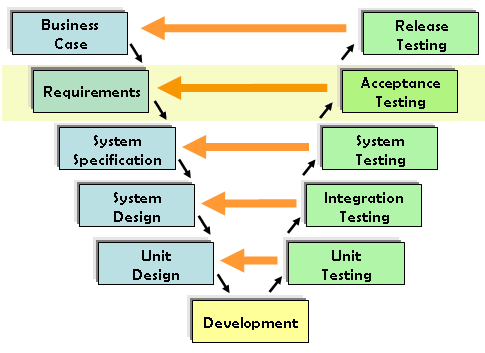
**Need of User Acceptance Testing:**

Need of User Acceptance Testing arises once software has undergone Unit, Integration and System testing because developers might have built software based on requirements document by their own understanding and further required changes during development may not be effectively communicated to them, so for testing whether the final product is accepted by client/end-user, user acceptance testing is needed.

* Developers code software based on requirements document which is their “own” understanding of the requirements and **may not actually be what the client needs from the software**.
* Requirements changes during the course of the project may not be communicated effectively to the developers.

**Acceptance Testing and V-Model**

In VModel, User acceptance testing corresponds to the requirement phase of the Software Development life cycle(SDLC).



**How to execute UAT Tests:**

UAT is done by the intended users of the system or software. This type of Software Testing usually happens at the client location which is known as Beta Testing. Once Entry criteria for UAT are satisfied, following are the tasks need to be performed by the testers:

* Analysis of Business Requirements
* Creation of UAT test plan
* Identify Test Scenarios
* Create UAT Test Cases
* Preparation of Test Data(Production like Data)
* Run the Test cases
* Record the Results
* Confirm business objectives

**Features:**

* All-in-one UAT solution
* Works across all ERPs and applications
* Test any process end-to-end
* Automatically capture everything
* Train and use within 30 minutes
* Instant test result notifications
* Easy annotations & comments

**Example Guidelines for UAT:**

* Most of the times in regular software developing scenarios, UAT is carried out in the QA environment. If there is no staging or UAT environment
* UAT is classified into Beta and Beta and Alpha testing but it is not so important when software is developed for a service based industry
* UAT makes more sense when the customer is involved to a greater extent

4. Integration Testing: Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

[Unit testing](https://www.javatpoint.com/unit-testing) uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

**Example of Integration Test Case**

Integration[Test Case](https://www.guru99.com/test-case.html)differs from other test cases in the sense it focuses mainly on the interfaces & flow of data/information between the modules. Here priority is to be given for the integrating links rather than the unit functions which are already tested.

Sample Integration Test Cases for the following scenario: Application has 3 modules say ‘Login Page’, ‘Mailbox’ and ‘Delete emails’ and each of them is integrated logically.

Here do not concentrate much on the Login Page testing as it’s already been done in [Unit Testing](https://www.guru99.com/unit-testing-guide.html). But check how it’s linked to the Mail Box Page.

**Types of Integration Testing:**

Software Engineering defines variety of strategies to execute Integration testing, viz.

* Big Bang Approach :
* Incremental Approach: which is further divided into the following
  + Top Down Approach
  + Bottom Up Approach
  + Sandwich Approach – Combination of Top Down and Bottom Up

**5.Regression Testing:** Regression testing is a type of software testing conducted after a code update to ensure that the update introduced no new bugs. This is because new code may bring in new logic that conflicts with the existing code, leading to defects. Usually, QA teams have a series of regression test cases for important features that they will re-execute each time these code changes occur to save time and maximize test efficiency.

**Examples Of Regression Testing:**

Keep in mind that regression testing is a practice and method. There isn’t a testing tool for regression testing. Any test that was developed at the initial launch of a certain feature, brought up to run at later releases, is already a regression test. You can have regression tests by application attribute and type:

* **Attribute**: visual, functional, performance, security
* **Application layer**: UI and API
* **Application type**: web, mobile, API and desktop
* **Granularity**: end-to-end, integration and unit tests (test pyramid)

To perform regular check ups on existing code, regression testing basically involves:

* Testers writing and building automated test suites
* Testers and business analysts filtering out release-specific tests to re-run on affected areas from new releases

Let’s consider a simple example. We have a community-type mobile application where users can share their own insights in the form of short posts for other users to comment on and interact with. To further improve UX, a software engineer wants to add a personalized post recommendation feature based on users’ interests and past activity.

After the new feature is written, regression testing is run to make sure that the new recommendation feature does not affect the existing feature. The test plan might involve running manual or automated regression tests to check the functionality of the existing code and if there are any conflicts between 2 versions. If any bugs are discovered, they are fixed and the regression test is run again until all tests pass.

At a larger scale, companies whose business model revolves around their digital products need regression testing to frequently check up on their core features. To improve productivity in regression testing, automation testing tools are commonly employed.

**Why Regression Testing?**

Regression testing process is essential in the testing scope. As it can identify if code changes or enhancements are introducing new defects or disrupting existing functional tests.

Without a regression testing process, even minor code changes may have a chance of leading to costly errors. It is, therefore, a systematic practice to help maintain software quality. This method helps prevent the recurrence of known issues and boosts confidence in the software.



**6. Performance Testing:** Performance testing is the practice of evaluating how a system performs in terms of responsiveness and stability under a particular workload. Performance tests are typically executed to examine speed, robustness, reliability, and application size. The focus of Performance Testing is checking a software program’s

* **Speed** – Determines whether the application responds quickly
* **Scalability** – Determines the maximum user load the software application can handle.
* **Stability** – Determines if the application is stable under varying loads

**When we use performance testing?**

We will do performance testing once the software is stable and moved to the production, and it may be accessed by the multiple users concurrently, due to this reason, some performance issues may occur. To avoid these performance issues, the tester performs one round of performance testing.

Since it is non-functional testing which doesn't mean that we always use performance testing, we only go for performance testing when the application is functionally stable.

**Types of Performance Testing:**

There are primarily six types of performance testing in software testing, which are explained below.

* **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**](https://www.guru99.com/stress-testing-tutorial.html)**–** 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.

**Performance testing example:**

Let us take one example where we will test the behavior of an application where the desired load is either less than 1000 or equal to 1000 users.

In the below image, we can see that the 100 up users are increased continuously to check the maximum load, which is also called upward scalability testing.

**Scenario1:** When we have the 1000 users as desired load, and the 2.7/sec is goal time, these scenarios will pass while performing the load test because in load testing, we will concentrate on the no. of users, and as per the requirement it is equal to 1000 user.

**Scenario2:** In the next scenario, we will increase the desired load by 100 users, and goal time will go up to 3.5\sec. This scenario will pass if we perform stress testing because here, the actual load is greater than (1100) the desired load (1000).

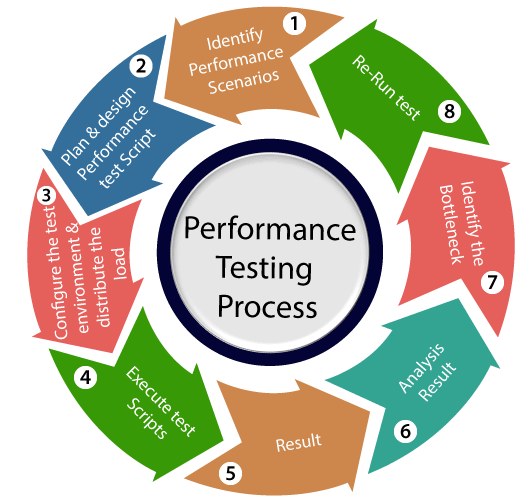
**Scenario3:** In this, if we increase the desired load three times as

1200 → 3.5\sec: [it is not less than or equal to the desired load that's why it will Fail]

1300 → 4\sec: [it is not less than or equal to the desired load. i.e., Fail]

1400 → Crashed

**Performance testing process:**

****

In [Software Engineering](https://www.guru99.com/what-is-software-engineering.html), Performance testing is necessary before marketing any software product. It ensures customer satisfaction & protects an investor’s investment against product failure. Costs of performance testing are usually more than made up for with improved customer satisfaction, loyalty, and retention.

**7. Load Testing:** It 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.

**Need of Load Testing:**

Some extremely popular sites have suffered serious downtimes when they get massive traffic volumes. E-commerce websites invest heavily in advertising campaigns, but not in Load Testing to ensure optimal system performance, when that marketing brings in traffic.

**Goals of Load Testing:**

Loading testing identifies the following problems before moving the application to market or Production:

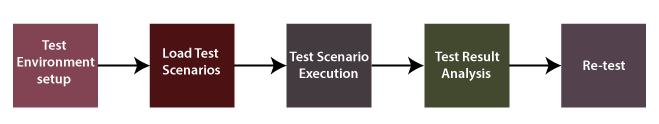
* Response time for each transaction
* Performance of System components under various loads
* Performance of Database components under different loads
* Network delay between the client and the server
* Software design issues
* Server configuration issues like a Web server, application server, database server etc.
* Hardware limitation issues like CPU maximization, memory limitations, network bottleneck, etc.

**Rules for load testing:**

During the execution of the load testing, a test engineer should follow the below rules:

* A test engineer tries to evade downloading images on the site.
* Once the application becomes functionally stable, load testing should be planned.
* The reliability of response time concludes the past period should be logged and the same should be compared with several test runs.
* For each scenario or script number of users should be decided.

**Load Testing Process:**



Load testing is defined as a type of software testing that determines a system’s performance under real-life load conditions. load testing typically improves performance bottlenecks, scalability and stability of the application before it is available for production.

**8. Security Testing:** Security testing is an integral part of software testing, which is used to discover the weaknesses, risks, or threats in the software application and also help us to stop the nasty attack from the outsiders and make sure the security of our software applications.The primary objective of security testing is to find all the potential ambiguities and vulnerabilities of the application so that the software does not stop working. If we perform security testing, then it helps us to identify all the possible security threats and also help the programmer to fix those errors.

It is a testing procedure, which is used to define that the data will be safe and also continue the working process of the software.

**Types of Security Testing in Software Testing:**

There are seven main types of security testing as per Open Source Security Testing methodology manual. They are explained as follows:

* **Vulnerability Scanning**: This is done through automated software to scan a system against known vulnerability signatures.
* **Security Scanning:** It involves identifying network and system weaknesses, and later provides solutions for reducing these risks. This scanning can be performed for both Manual and Automated scanning.
* **Penetration testing**: This kind of testing simulates an attack from a malicious hacker. This testing involves analysis of a particular system to check for potential vulnerabilities to an external hacking attempt.
* **Risk Assessment:** This testing involves analysis of security risks observed in the organization. Risks are classified as Low, Medium and High. This testing recommends controls and measures to reduce the risk.
* **Security Auditing:** This is an internal inspection of Applications and [Operating systems](https://www.guru99.com/os-tutorial.html) for security flaws. An audit can also be done via line by line inspection of code
* **Ethical hacking:** It’s hacking an Organization Software systems. Unlike malicious hackers, who steal for their own gains, the intent is to expose security flaws in the system.
* **Posture Assessment:** This combines Security scanning,[Ethical Hacking](https://www.guru99.com/ethical-hacking-tutorials.html)and Risk Assessments to show an overall security posture of an organization.

**How we perform security testing?**

The security testing is needed to be done in the initial stages of the software development life cycle because if we perform security testing after the software execution stage and the deployment stage of the SDLC, it will cost us more.Now let us understand how we perform security testing parallel in each stage of the software development life cycle(SDLC).

**Example of security testing:**

Generally, the type of security testing includes the problematic steps based on overthinking, but sometimes the simple tests will help us to uncover the most significant security threats.

Let us see a sample example to understand how we do security testing on a web application:

* Firstly, log in to the web application.
* And then log out of the web application.
* Then click the BACK button of the browser to verify that it was asking us to log in again, or we are already logged-in the application.

For an application or the software, it is necessary to perform security testing to verify that the sensitive information is still private. In software testing, the security testing is essential because it helps us to save our necessary data ultimately. In this, the test engineer will act as an invader and test the system or detect the security defects.

**9. Portability Testing:** Portability testing is a process of testing with ease with which the software or product can be moved from one environment to another. It is measured in terms of maximum amount of effort required to transfer from one system to another system.

The portability testing is performed regularly throughout the software development life cycle in an iterative and incremental manner.

**Types of Portability Testing:**

* **Hardware Portability Testing:**This testing ensures that the software can function on various hardware setups. It requires testing the application across a range of hardware architectures, processors, and devices.
* **Browser portability Testing:** It tests for browser portability. Testers confirm that the software runs well and shows consistently in popular browsers such as Internet Explorer, Firefox, Chrome, and Safari.
* **Data Portability Testing:** Data portability testing evaluates the program’s capacity to move data between various contexts and systems. It guarantees that throughout transitions, data can be imported and exported without modification or loss.
* **Middleware Portability Testing:** Testing for middleware portability assesses how well a piece of software works with various middleware services and components. It guarantees that a variety of middleware technologies can be seamlessly integrated with the application.
* **Installation and Uninstallation Portability Testing:**This kind of testing evaluates how simple it is to install and remove the software in various settings. It verifies if the installation procedure is dependable and consistent across a range of platforms.

**Portability Testing Attributes:**

* **Adaptability:** Adaptability is defined as the capacity of the software application to adopt to a particular environment without any effort. Common communication standards between multiple systems help in enhancing the adaptability of the system as a whole.
* **Installability:** Installability is defined as the capacity of software application to get installed to the desired environment without using extra resources. Installability is performed on a software that is to be installed in a target environment.
* **Replaceability:** Replaceability is defined as the capacity of software application to replace other software in a particular environment. The software application that is replacing the previous software application must produce the same results on all target environments.
* **Co-existence:** Co-existence is defined as the capacity of software application to work with other software application on the system without disturbing each other and sharing the same resource. Specially this testing is used in large systems which include multiple subsystems as part of it.

10. Accountability Testing: Accountability testing in software refers to the process of evaluating a software application's ability to track and record actions, events, or changes made within the system. It ensures that the software accurately captures and logs relevant information to hold users or system components accountable for their actions. This type of testing is particularly important in systems where auditing, compliance, or regulatory requirements need to be met.

Key aspects of accountability testing may include:

1. **Logging Mechanisms**: Testing the effectiveness of the logging mechanisms within the software to ensure they capture relevant events, actions, and changes.
2. **Data Integrity**: Verifying that the logged information accurately reflects the actions taken within the system and maintains data integrity.
3. **Access Controls**: Assessing the software's ability to enforce access controls and permissions to ensure that only authorized users can perform certain actions, and that these actions are appropriately logged.
4. **Audit Trails**: Evaluating the completeness and accuracy of audit trails generated by the software, which provide a chronological record of system activities.
5. **Compliance Requirements**: Ensuring that the software meets specific compliance requirements or standards related to accountability, such as HIPAA (Health Insurance Portability and Accountability Act) in healthcare or GDPR (General Data Protection Regulation) in data privacy.
6. **Error Handling**: Testing the software's ability to handle errors related to logging and accountability, such as failures in recording events or insufficient storage capacity for logs.

Overall, accountability testing helps organizations maintain transparency, traceability, and compliance within their software systems by ensuring that actions taken within the system are appropriately tracked, recorded, and attributable to the responsible parties.

**11. Reliability Testing:** Reliability Testing is a software testing process that checks whether the software can perform a failure-free operation in a particular environment for a specified time period. The purpose of Reliability testing is to assure that the software product is bug-free and reliable enough for its expected purpose.

Reliability means “yielding the same,” in other terms, the word “reliable” means something is dependable and that it will give the same outcome every time. The same is true for Reliability testing.

**Reliability Testing Example:**

The probability that a PC in a store is up and running for eight hours without crashing is 99%; this is referred to as reliability.

Reliability Testing can be categorized into three segments,

* Modeling
* Measurement
* Improvement

**Different Ways to Perform Reliability Testing:**

1. **Stress testing:** This testing involves subjecting the system to high levels of load or usage to identify performance bottlenecks or issues that can cause the system to fail
2. **Endurance testing:** Endurance testing involves running the system continuously for an extended period to identify issues that may occur over time
3. **Recovery testing:**Recovery testing is testing the system’s ability to recover from failures or crashes.
4. **Environmental Testing:**Conducting tests on the product or system in various environmental settings, such as temperature shifts, humidity levels, vibration exposure or shock exposure, helps in evaluating its dependability in real-world circumstances.
5. **Performance Testing:** It is possible to make sure that the system continuously satisfies the necessary specifications and performance criteria by assessing its performance at both peak and normal load levels.
6. **Regression Testing:** After every update or modification, the system should be tested again using the same set of test cases to help find any potential problems caused by code changes.
7. **Fault Tree Analysis:** Understanding the elements that lead to system failures can be achieved by identifying probable failure modes and examining the connections between them.

It is important to note that reliability testing may require specialized tools and test environments, and that it’s often a costly and time-consuming process.

**Objective of Reliability Testing:**

1. To find the perpetual structure of repeating failures.
2. To find the number of failures occurring is the specific period of time.
3. To discover the main cause of failure.
4. To conduct performance testing of various modules of software product after fixing defects.
5. It builds confidence in the market, stakeholders and users by providing a dependable product that meets quality criteria and operates as expected.
6. Understanding the dependability characteristics and potential mechanisms of failure of the system helps companies plan and schedule maintenance actions more efficiently.
7. It evaluates whether a system or product can be used continuously without experiencing a major loss in dependability, performance or safety.
8. It confirms that in the absence of unexpected shutdown or degradation, the system or product maintains constant performance levels under typical operating settings.

**Types of reliability Testing:**

Software reliability testing includes Feature Testing, Load Testing, and Regression Testing

**Load Testing:-**

Usually, the software will perform better at the beginning of the process, and after that, it will start degrading. Load Testing is conducted to check the performance of the software under the maximum workload.

**Regression Testing :-**

Regression testing is mainly used to check whether any new bugs have been introduced because of fixing previous bugs. Regression Testing is conducted after every change or update of the software features and their functionalities.

**Importance of Reliability Testing:**

A thorough assessment of reliability is required to improve the performance of software products and processes. Testing software reliability will help software managers and practitioners to a great extent.

To check the reliability of the software via testing:-

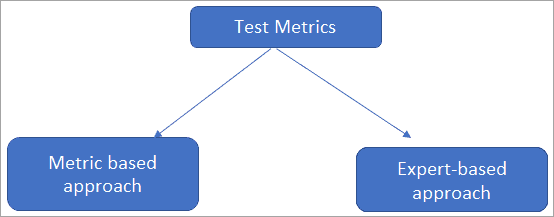
1. A large number of test cases should be executed for an extended period to determine how long the software will execute without failure.
2. The test case distribution should match the software’s actual or planned operational profile. The more often a function of the software is executed, the greater the percentage of test cases that should be allocated to that function or subset.

Reliability Testing is an important part of a reliability engineering program. More correctly, it is the soul of a reliability engineering program. Furthermore, reliability tests are mainly designed to uncover particular failure modes and other problems during software testing.

12. Efficiency Testing: Efficiency testing test the amount of code and testing resources required by a program to perform a particular function. Software Test Efficiency is number of test cases executed divided by unit of time (generally per hour).It is internal in the organization how much resources were consumed how much of these resources were utilized.

**Techniques Used For Test Efficiency:**

Both the techniques, given below, can be used for evaluating test efficiency:



**Example Of Test Efficiency:**

**#1)** To launch software that is of high quality i.e. bug-free and is to be delivered on time.

To make the above expectation successful, the team must focus on efficiency i.e.

* Customer requirement to be fulfilled.
* To verify the number of resources allocated to the project and the actual number of resources utilized.
* The tools being used are the latest to increase efficiency.
* Team members being utilized are highly skilled.

**#2)** To test a form that has the validation of 10 characters on Name, Surname/City fields.

The tester can automate to test the form. The file with the number of inputs where Name/Surname/City details are mentioned with blanks, characters between 1-10, characters more than 10, spaces between the characters, special characters, numbers only, caps, small characters, etc can be created.

**#3)** To test a login page.

The tester can get the data for username and password with multiple scenarios such as correct username/incorrect password, correct username/correct password, Incorrect user/correct password, incorrect user/incorrect password, etc.

The list can be populated through SQL injections. Automation allows the tester to test more scenarios in less time. The tester themselves can decide the best technique to execute cases to increase efficiency.

**Disadvantages**

* To fulfill the metrics requirement, out of box thinking & creativity of the tester, and exploration testing can be hampered as the focus would remain to work as per the metrics only.
* The focus moves toward documentation rather than performing testing that results in inefficiency.
* Sometimes filing the metrics on a regular basis creates demotivation in the resources.

**Advantages**

* Test Metrics improve the productivity of the resources – as defining the metrics gives a clear objective to the tester.
* It improves the tracking system. Maintaining the metric helps to track the testing activities and progress.
* The testing efforts can be easily visible.
* The testing team can provide their efficiency anytime if asked for.

**13. 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 analyze 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 behavior can be studied when exposed to a high volume of data.

For example, testing the music site behavior 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

**Why to do Volume Testing:**

The objective of performing the volume testing is to

* Check system performance with increasing volumes of data in the database
* To identify the problem that are likely to occur with large amount of data
* To figure out the point at which the stability of the system degrades
* Volume Testing will help to identify the capacity of the system or application – normal and heavy volume

**Volume Testing Attributes:**

Following are the important attributes that are checked during the volume testing:

* **System’s Response Time:** During the volume testing, the response time of the system or the application is tested. It is also tested whether the system responses within the finite time or not. If the response time is large then the system is redesigned.
* **Data Loss:** During the volume testing, it is also tested that there is no data loss. If there is data loss some key information might be missing.
* **Data Storage:** During the volume testing, it is also tested that the data is stored correctly or not. If the data is not stored correctly then it is restored accordingly in proper place.
* **Data Overwriting:** In volume testing, it is tested that whether the data is overwritten without giving prior information to the developer. If it so then developer is notified.

**Advantages of Volume Testing:**

* Volume testing is helpful in saving maintenance cost that will be spent on application maintenance.
* Volume testing is also helpful in a rapid start for scalability plans.
* Volume testing also helps in early identification of bottlenecks.
* Volume testing ensures that the system is capable of real world usage.

**Disadvantages of Volume Testing:**

* More number of skilled resources are needed to carry out this testing.
* It is sometimes difficult to prepare test cases with respect to the number of volume of data to be tested.
* It is a time consuming technique since it requires lot of time to decide the number of volume of data and test scenarios.
* It is a bit costly as compared to another testing technique.
* It is not possible to have the exact break down of memory used in the real world application.

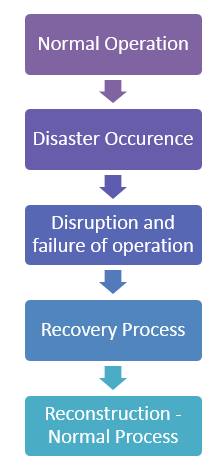
**14. Recovery Testing:** Recovery Testing is software testing technique which verifies software’s ability to recover from failures like software/hardware crashes, network failures etc. The purpose of Recovery Testing is to determine whether software operations can be continued after disaster or integrity loss. Recovery testing involves reverting back software to the point where integrity was known and reprocessing transactions to the failure point.

**Recovery Testing Example:**

When an application is receiving data from the network, unplug the connecting cable.

* After some time, plug the cable back in and analyze the application’s ability to continue receiving data from the point at which the network connection was broken.
* Restart the system while a browser has a definite number of sessions open and check whether the browser is able to recover all of them or not

**Life Cycle of Recovery Process:**



**How to do Recovery Testing?**

While performing recovery testing following things should be considered.

* We must create a test bed as close to actual conditions of deployment as possible. Changes in interfacing, protocol, firmware, hardware, and software should be as close to the actual condition as possible if not the same condition.
* Through exhaustive testing may be time-consuming and a costly affair, identical configuration, and complete check should be performed.
* If possible, testing should be performed on the hardware we are finally going to restore. This is especially true if we are restoring to a different machine than the one that created the backup.
* Some backup systems expect the hard drive to be exactly the same size as the one the backup was taken from.
* Obsolescence should be managed as drive technology is advancing at a fast pace, and old drive may not be compatible with the new one. One way to handle the problem is to restore to a [virtual machine](https://www.guru99.com/best-virtual-machine-software.html). Virtualization software vendors like VMware Inc. can configure virtual machines to mimic existing hardware, including disk sizes and other configurations.
* Online backup systems are not an exception for testing. Most online backup service providers protect us from being directly exposed to media problems by the way they use fault-tolerant storage systems.
* While online backup systems are extremely reliable, we must test the restore side of the system to make sure there are no problems with the retrieval functionality, security or encryption.

Recovery Testing improves the quality of the system by eliminating the potential flaws in the system so that the system works as expected.

15.Responsive Testing: [Responsive testing](https://www.lambdatest.com/blog/responsive-testing-of-a-locally-hosted-website-a-complete-guide/) ensures that users have the best experience with your site, regardless of their device. The goal of testing responsive websites is to ensure a seamless experience across different digital devices. In this day and age, we live in a world where technology has enabled convenience, and we are now dependent on our devices to function.

Because of the growing market for mobile devices, businesses are developing strategies to create user-friendly websites. They use mobile-first design, progressive web apps, single-page applications, and more. However, for a unified user experience across devices and platforms, we need to consider screen resolutions and device capabilities.

**Why is Responsive Testing Important?**

Responsive testing of web apps is crucial at every stage of development to ensure that the end-user requirements are met. Here are the following reasons highlighting the importance to test responsive websites:

* **Plethora of Devices, OS & Browsers:** To ensure that your site’s content is available to all visitors, verification of the content needs to be done for different screen-sized mobiles, operating systems, and browsers. While a site designed in one browser may appear as intended in another browser, it should not be assumed that this is necessarily the case.
* **Need for Robustness:** It is crucial to ensure that the website is loading at the same speed on different devices and browsers so that users do not become frustrated by lagging or timed-out content. If a website loads slowly or doesn’t display correctly on mobile devices, users will have a poor experience. Therefore, testing a website’s performance is essential for ensuring users have a positive experience on mobile-responsive websites.
* **Website Navigation:** When testing a mobile website, one of the most common defects found is that pages don’t load as expected when navigated among the site’s links. It also happens that links are missing, images are not loaded, or timed out while playing with navigation.
* **Multiple Images and Videos:** When creating a responsive website, it is essential to test whether all types of images and videos are displayed as expected on different phones, browsers, etc. Sometimes some videos play well on Android, but they don’t even load on iOS, or some images appear broken on some versions of a mobile operating system while they are perfect on others. Such issues will give a terrible impression if testing is not **done correctly.**

**Types of Responsive Website Testing:**

There are several types of responsiveness testing types that can be performed to ensure that a website is responsive and functions correctly on a variety of devices and screen sizes:

* **Visual Regression Testing** Visual regression testing is a part of regression testing that involves taking screenshots of a website on different devices and comparing them to ensure that the layout and design are consistent across all screens.
* **Visual Layout Testing:** Visual layout testing tools allow users to check that the website's layout adjusts correctly to different screen sizes and orientations and that all content is displayed correctly and is easily readable and navigable.
* **Cross browser testing:** Cross browser compatibility testing is the most significant kind of front end testing. Testers can determine if a website functions as intended when viewed using various browsers/devices/OS combinations. In addition, cross browser testing makes it possible for people to experience the same thing across multiple browsers.
* **Functional Testing:** This involves testing the website's functionality on different devices to ensure that all features and interactions work as expected. Functional testing evaluates the various functions of the application. It checks the user interface, database, APIs, client/server communication, security, and other components.
* **Performance Testing:** Performance testing assesses a product's quality and capability under varying workloads. Performance testing ensures that the system performs adequately, reliably, and with stability. This involves testing the website's performance on different devices and networks to ensure that it loads quickly and runs smoothly.
* **Usability Testing:** Usability testing is a technique for evaluating the user experience of a web product or service by testing it with users. This involves testing the website's usability on different devices to ensure it is easy for users to navigate and use.

**Advantages of Responsive Testing:**

Responsive web test is an integral part while delivering high-quality products. There are several advantages to performing website responsive testing:

* **Improved User Experience**: It is important to ensure that a website is fully responsive to ensure that all users, regardless of their device, have a positive and seamless experience when interacting with the website.
* **Increased Accessibility:** A responsive website can be accessed and used by a broader range of devices and screen sizes, which can help to expand its reach and accessibility.
* **Enhanced Search Engine Optimization:** Google's search algorithms give higher rankings to mobile-friendly websites, so having a responsive website can help to improve a website's search engine ranking.
* **Cost Saving:** Developing and maintaining a separate mobile website can be time-consuming and costly. A responsive website can save time and money by eliminating the need to create and maintain a different mobile version.
* **Improved Conversion Rates:** A responsive website can improve conversion rates by providing a consistent user experience across all devices, which helps to build trust and credibility with users.

**16. Visual Testing:** Visual Testing is also called Visual UI Testing. It validates whether the developed software user interface (UI) is compatible with the user’s view. It ensures that the developed web design is correctly following the spaces, sizes, shapes, and positions of UI elements. It also ensures that the elements are working properly with various devices and browsers. Visual testing validates how multiple devices, browsers, operating systems, etc., affect the software.

**Features of Visual Testing:**

* **Testing of User Interface (UI) Elements:** Visual testing confirms that all user interface (UI) components, including text fields, buttons, icons, images and other graphical elements, are presented accurately and positioned in accordance with the design.
* **Arrangement and Coherence of Design:**It guarantees that the layout of the programme remains the same on various screens, sizes, and resolutions. To keep a design solid, elements must be positioned and aligned correctly.
* **Verification of Theme and Color:** Visual testing verifies that the application’s color scheme and theme comply to the design guidelines.
* **Testing Text and Fonts:**Verifying that fonts are presented correctly, that the content is readable and that there are no problems with font size, style, or formatting are all included in this aspect.

**Working of Visual Testing:**

Visual tests generate, compare and analyze browser snapshots to detect if any pixels have changed. These pixel’s differences are called visual pixels.

**Steps in Visual Testing:**

* The Quality Analyst or the tester runs the developed code to test the web application’s user interface part.
* Initially, it will record the screen as snapshots. It acts as a **baseline** with which the further test results will get compared.
* After that, the QA runs the code in the background and it will take or record the snapshots of those running codes.
* Now, it will start comparing with the baseline snapshots.
* If changes are found among those snapshots then the test is considered as failed.
* If no changes are found then it will be tested positively.

**Why Visual Testing?**

Visual testing is done because visual errors happen more frequently than one might realize. Some of the reasons for doing visual testing are-

* It verifies or ensures that the developed product UI appears as expected to the users.
* It helps in evaluating the defects in the UI interface.
* It correctly detects the variations in the UI which is not relevant to the baseline snapshots.
* It helps to create dedicated visual test cases and covers the functional points.
* Visual testing allows the tester or Quality Analyst to evaluate the test cases visually which is easier to carry out.

**Advantages of Visual Testing:**

1. Increased quality of code and UI.
2. Rapid bug detection.
3. Control of Views on different devices and browsers.
4. Easy to implement.
5. Fast implementation.
6. Automation.
7. Reduction of Code.
8. Efficient feedback of results during testing.

**Disadvantages of Visual Testing:**

1. It can not detect smaller defects.
2. It requires the minimum skills to implement. It is a little difficult for beginners.
3. Sometimes, video recording is not so clear.
4. Possibilities of containing the detect even though testing has been implemented.
5. Tests only the visible or light surface.

Visual testing concentrates on an application’s user interface and visual components, is an essential component of the software testing process. This testing technique is crucial to preserving the overall quality of the software, discovering discrepancies in the design, and guaranteeing a satisfactory user experience. It is not only a best practice but also a deliberate approach to provide software that satisfies functional and aesthetic criteria, increasing user satisfaction and fostering a positive brand perception. Visual testing should be incorporated into the development lifecycle.