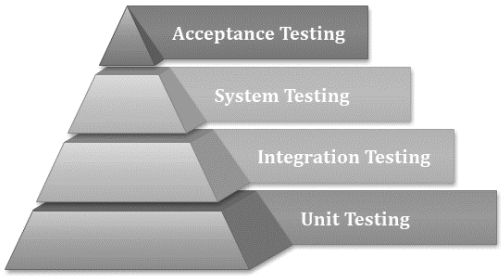

UNIT TESTING

In software engineering, unit testing is a level of software testing in which individual units/components are tested.

A unit is a smallest testable part/module of any software application. In procedural programming, a unit is an individual program, function, procedure. In object-oriented programming, a unit may be a method.

Unit testing of the software product is done during the software development stage saves time and money in the end.

If unit testing is skipped by the software developers then there is higher chances of defects during the integration, system, acceptance and also beta testing.

The goal of unit testing is to isolate each part of source code and verifies that each part works properly as designed and to check that source code meets the requirements and gives the expected output.

The main advantage of unit testing is that it identify the problem earlier in the application and if any issue occurs then it can be fix before integrate the units.

In SDLC, STLC, V Model, unit testing is the first level of software testing done before integration testing.

Objective of unit testing :

Unit testing separate a section of code.

To verify that source code performs as designed in an expected manner.

This testing is used to test every function and procedure individually.

To fix defects early in software development stage.

Software developers performs this testing to save time and cost both.

Unit testing helps the developers to understand the code base and enable them to make changes quickly.

To help for code reuse.

Workflow of Unit Testing:

In order to do Unit Testing, developers write a section of code to test a specific function in software application. Developers can also isolate this function to test more rigorously which

reveals unnecessary dependencies between function being tested and other units so the dependencies can be eliminated. Developers generally use UnitTest framework to develop automated test cases for unit testing.

Unit Testing is of two types

Manual

Automated

Unit testing is commonly automated but may still be performed manually. Software Engineering does not favor one over the other but automation is preferred. A manual approach to unit testing may employ a step-by-step instructional document.

Under the automated approach-

A developer writes a section of code in the application just to test the function. They would later comment out and finally remove the test code when the application is deployed.

A developer could also isolate the function to test it more rigorously. This is a more thorough unit testing practice that involves copy and paste of code to its own testing environment than its natural environment. Isolating the code helps in revealing unnecessary dependencies between the code being tested and other units or data spaces in the product. These dependencies can then be eliminated.

A coder generally uses a UnitTest Framework to develop automated test cases. Using an automation framework, the developer codes criteria into the test to verify the correctness of the code. During execution of the test cases, the framework logs failing test cases. Many frameworks will also automatically flag and report, in summary, these failed test cases. Depending on the severity of a failure, the framework may halt subsequent testing.

The workflow of Unit Testing is 1) Create Test Cases 2) Review/Rework 3) Baseline 4) Execute Test Cases.

Advantages of Unit Testing:

Unit testing allows the programmer to improve the code and make sure that the components works properly in the expected manner.

Unit testing enables to verify units of an application without waiting for others to be completed due to the modular nature of this testing.

This testing reduces bugs in the newly developed features or when altering the existing working of an application.

It helps to maintain the code.

This testing is used to reducing the cost of defect fixes because defects are found early in the software development stage.

Fixing defects in unit testing can fix many other defects arising in later development.

Due to decreased defect count overall development time can be saved to complete a project.

Disadvantages of Unit Testing:

Unit testing can not be expected to find all defects/errors present in a program/code.

Unit testing is not able to find integration level errors or system level errors because it only focuses on components of the code/program.

This testing can be time-consuming.

INTEGRATION TESTING

Integration testing is a level of software testing in which units/components are integrated in a single module and then test to check communication between these components.

Integration Testing is the second level of software testing which is done after unit testing and before system testing. When the integration testing is performed on the modules, system testing is performed.

This testing comes under both black box and white box testing. This testing is done by software testers or developers.

This testing is also known as 'I & T' (Integration and Testing), 'String Testing' and 'Thread Testing'. This testing is an extension of unit testing.

The term 'integrate' refers to combine two or more things in a single thing to become more effective . The term defines the concept of integration testing.

The main goal of integration testing is to test the functional, performance, and reliability between the components that are integrated.

The modules tested individually in unit testing then integrate them one by one until all the modules are integrated and check the flow of data between the modules.

The test cases of integration testing differs from other test cases i.e. it only focuses on the interfaces & flow of the data between the modules.

Objective of Integration Testing :

To reduce risk in system testing

Integration Testing is performed to verify the software modules work correctly.

If client alters the requirements then integration testing is necessary because new requirements may not be tested.

Sometimes, modules communicate with third party APIs or tools so it is necessary to test that data accepted by that tool/API is correct and generates expected response or not.

If developer deploys the change in requirements and not perform unit testing on them then integration testing is necessary.

This testing is necessary to build confidence in the quality of the interfaces.

To find issues/bugs if present in the interface or components or systems.

Guidelines for Integration Testing

We go for the integration testing only after the functional testing is completed on each module of the application.

We always do integration testing by picking module by module so that a proper sequence is followed, and also we don't miss out on any integration scenarios.

First, determine the test case strategy through which executable test cases can be prepared according to test data.

Examine the structure and architecture of the application and identify the crucial modules to test them first and also identify all possible scenarios.

Design test cases to verify each interface in detail.

Choose input data for test case execution. Input data plays a significant role in testing.

If we find any bugs then communicate the bug reports to developers and fix defects and retest.

Perform positive and negative integration testing.

Here positive testing implies that if the total balance is Rs15, 000 and we are transferring Rs1500 and checking if the amount transfer works fine. If it does, then the test would be a pass.

And negative testing means, if the total balance is Rs15, 000 and we are transferring Rs20, 000 and check if amount transfer occurs or not, if it does not occur, the test is a pass. If it happens, then there is a bug in the code, and we will send it to the development team for fixing that bug.

Integration testing done using two approach

1) Big Bang Approach

2) Incremental Approach: which is further divided into the following

Top Down Approach

Bottom Up Approach

Sandwich Approach - Combination of Top Down and Bottom Up

Big Bang Testing

Big Bang Testing is an Integration testing approach in which all the components or modules are integrated together at once and then tested as a unit. This combined set of components is considered as an entity while testing. If all of the components in the unit are not completed, the integration process will not execute.

Incremental Testing

In the Incremental Testing approach, testing is done by integrating two or more modules that are logically related to each other and then tested for proper functioning of the application. Then the other related modules are integrated incrementally and the process continues until all the logically related modules are integrated and tested successfully.

Incremental Approach, in turn, is carried out by two different Methods:

Bottom Up- Bottom-up Integration Testing is a strategy in which the lower level modules are tested first. These tested modules are then further used to facilitate the testing of higher level

modules. The process continues until all modules at top level are tested. Once the lower level modules are tested and integrated, then the next level of modules are formed.

Top Down- Top Down Integration Testing is a method in which integration testing takes place from top to bottom following the control flow of software system. The higher level modules are tested first and then lower level modules are tested and integrated in order to check the software functionality. Stubs are used for testing if some modules are not ready.

Sandwich/Mixed Testing- it is a strategy in which top level modules are tested with lower level modules at the same time lower modules are integrated with top modules and tested as a system. It is a combination of Top-down and Bottom-up approaches therefore it is called Hybrid Integration Testing. It makes use of both stubs as well as drivers.

Advantages of Integration Testing :

This testing provides faster development and increased confident level.

It is easy to combine different modules.

Code coverage is high than other approaches.

It easily caught system level issues like mistakes in integration,broken database etc.

This testing can start when the relevant modules/components are available.It is not necessary to wait until all the modules are implemented and unit tested.

Integration testing can be used in the early as well as later stages of the testing process and defects find easily.

This testing makes sure that the combined modules works properly.

Disadvantages of Integration Testing :

To be precise the success of Integration Testing lies in the perfection of the test plan.

Test plan is made by humans one needs to serve for its inefficiencies.

If any scenario is not covered in the test plan then errors can be occurs in the system testing.

This testing can be time-consuming.

SYSTEM TESTING

System testing is a process of evaluating the integrated hardware and software system as a whole. System testing is a level of software testing in which a complete and fully integrated software product is test to verify that the system meets requirements and works as expected or not.

This testing comes under the black box type testing so only the working features are evaluated and there is no need to know about the internal structure or design of the code, internal path details. This testing is done from the user's point of view.

System Testing is performed by software testing team that is independent of the development team to measure the quality of the system.

System testing is performed in the context of a system requirement specification (SRS) or a functional requirement specifications (FRS) or both. This testing contains both functional and Non-Functional testing of an application.

This testing is done after completing the unit and integration testing and before acceptance testing.

Workflow of System Testing :

Prepare a test plan

Create test cases and test scripts

Choose test data

Execute test cases

Defect reporting and retesting

Repeat test cycle if required

Types of System Testing :

There are some system testing which is mentioned below :

Performance Testing

Load Testing

Security Testing

Scalability Testing

Reliability Testing

Usability Testing

Regression Testing

Interoperability Testing

Advantages of System Testing :

This testing includes end to end testing.

This testing verifies validation, verification, application architecture and business requirement.

System testing provides an effective environment which helps to understand the user perspective.

System testing prevents the defects which can occur when the system goes live.

It provides more reliable and efficient results.

This testing done proper evaluation of the system meeting the functional requirements.

Disadvantages of System Testing :

Test cases are difficult to design if requirements are not clear.

All properties of the system can not be tested.

Only limited coverage of application as a software tester cannot target specific code section.

ACCEPTANCE TESTING

Acceptance level is a level of software testing in which entire system is tested to evaluate that the system meets the requirements or and software product is acceptable for delivery to the customer.

Acceptance testing is the fourth and last level of software testing. This testing is demanded by the customers when the system testing is completed and before actual use of the software.

A quality assurance team perform acceptance testing to ensure the software application meets business requirements and end-user needs.

This testing returns either a pass or fail result. If result is fail it means flaw is present in the software product and software is not ready for use.

This is a black box testing approach in which only the functionality is evaluated to ensure that system meets specified acceptance criteria. Acceptance criteria are defined on the basis of the following attributes :-

Performance

Scalability

Usability

Availabilty

Timeliness

Data integrity

Documentation

Installability

Objectives of Acceptance Testing :

This test is performed to find the defects in the software product.

It ensures that how well the application is created.

The main goal is to ensure that system meets the business requirements.

Acceptance testing is done to remove or minimize the errors when system is in use.

It provides feedback which helps to improve the system performance.

Process of Acceptance Testing :

Create the test plan

Design the test cases

Execute test cases

compare the system with the requirements

obtain the result

Re-testing (if required)

Types of Acceptance Testing :

There are various types of acceptance testing which is mentioned below :

User acceptance Testing

Business acceptance Testing

Alpha Testing

Beta Testing

Advantages of Acceptance Testing :

This testing is done to establish confidence in the system for use.

It provides security in the system.

This testing is less expensive.

It minimize or remove bugs in the system when use.

It is easy for the user to define the requirements

Test execution can be automated.

This type of testing involves end users.

Disadvantages of Acceptance Testing :

Software Developers are not involve in this testing.

It have low test coverage.

No capture and replay possible in this testing.