Software Testing Methods :
1. White Box Testing
2. Black Box Testing
3. Grey Box Testing

1.White box testing-
-White box testing is done by coder because code knowledge is required.
-It is also called as code level testing/unit testing/clear box testing.
-In white box testing whenever coder complete his code writing, he checks or compile code then if any bug found code have to solve it
-coder cannot send code to tested without doing white box testing
-coder check or test mostly positive scenarios only.
-white box testing has purpose to test correctness and completeness of the program.

2.Black box testing-
-Black box testing is known system and function testing.
-This testing is done by tester.
-Overall functionality get checked in this type of testing.
-Tester check internal functionality depend upon external functionality.
Example-Tester check whenever data is sign module got entered and users press sign up button,this button is process to store entered data.Tester check whether the data is stored correctly or not.
So here internal functionality is storing of data and external functionality is filling up data in fields and submit buttons process.

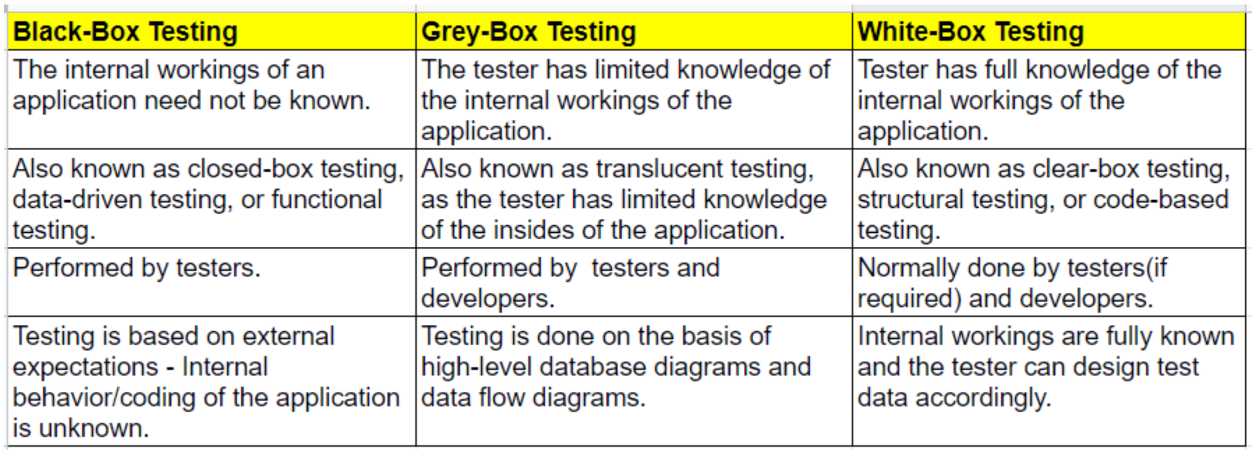
-Tester test the positive and negative scenarios.
Positive scenario means-
If suppose we have mobile number field with 10 digit functionality then as a tester we will check field functionality by entering 10 digit number whether it works or not.

Negative scenario means-

If suppose we have mobile number field with 10 digit functionality then as a tester if we check with 9 digits or less as it should not accept or more than 10 digits.

Grey box testing:
-Grey box testing is a combination of both white box and black box.
-To do grey box testing,tester need programming knowledge
-The role of grey box tester is whenever final software is handed over to tester tester check its functionality and if any fault occure in the output of function then tester try to solve that issue by self.So knowledge of coding is required.

Difference: -



Black-Box Testing	Grey-Box Testing	White-Box Testing
The internal workings of an application need not be known.	The tester has limited knowledge of the internal workings of the application.	Tester has full knowledge of the internal workings of the application.
Also known as closed-box testing, data-driven testing, or functional testing.	Also known as translucent testing, as the tester has limited knowledge of the insides of the application.	Also known as clear-box testing, structural testing, or code-based testing.
Performed by testers.	Performed by testers and developers.	Normally done by testers(if required) and developers.
Testing is based on external expectations - Internal behavior/coding of the application is unknown.	Testing is done on the basis of high-level database diagrams and data flow diagrams.	Internal workings are fully known and the tester can design test data accordingly.

<u>Levels of testing and types of testing..</u>

Testing Types are: Functional testing, regression testing, Smoke, Sanity, load and many more...

Levels Of Testing
1. Unit Testing
2. Integration Testing
3. System Testing
4. User Acceptance Testing (UAT)

1.Unit Testing (White box , Clear box , glass box , Structure based)

A Unit is a smallest testable piece of the software. It means testing a subprogram / module and checking is possible only by programmer.

Unit testing is a white box testing level and is performed at coding level

Method Used for unit testing: White Box Testing

When Unit testing should be done?

Testing can happen anytime when basic unit of code is ready

Unit testing should be done before Integration testing.

By whom unit testing should be done?

Unit testing should be done by the developers and testers (if required).

Unit Testing Techniques:

White Box Testing Techniques/The way which Unit Testing is performed :

- Statement Coverage - This technique is aimed at exercising all programming statements with minimal tests.
- Branch Coverage - This technique is running a series of tests to ensure that all branches are tested at least once.
- Path Coverage - This technique corresponds to testing all possible paths which means that each statement and branch is covered.

Unit testing tools available in the market, which are as follows:

- NUnit

- JUnit
- PHPunit
- Parasoft Jtest
- EMMA

How to achieve the best result via Unit testing?

- Naming conventions for unit test cases must be clear and consistent.
- Identified bugs on unit testing must be fixed before next level of SDLC
- Only one code should be tested at one time.
- If there are changes in the code of any module, ensure the unit test is available or not for that module.

Advantages

- Unit testing uses a module approach due to that any part can be tested without waiting for completion of another parts testing.
- The developing team focuses on the functionality of the unit.
- Refactoring of code can be possible after a number of days by dev and ensure the module is still working without any defect.

Disadvantages

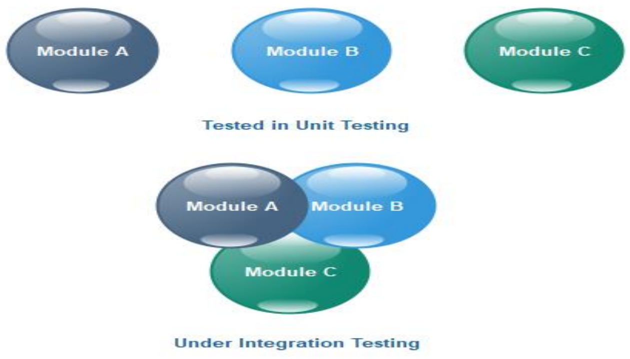
- It cannot identify integration error
- Some Code knowledge required

2. Integration Testing :

-In this phase of testing, individual modules are combined and tested as a group.

-Data transfer between the modules is tested thoroughly.

-Done by testers



-Integration testing is the process to check correctness and completeness of the flow of functionality whenever integration of module performed.

-We can say by two ways integration can be done-

1. Front end Integration-Here developer connect modules using “called functions”

Module1---(called-stub)----→module2

2. Back end integration-Here two tables in database connects each other by “Join” query.

Table 1----(Join)----Table 2

-Communication between 2 modules proceed through xml.

-Example of Integration Testing

1. UI + DataBase
2. Registration page and login page integrated with email and SMS after registration.

<u>Types of Integration Testing</u>

Integration testing can be classified into two parts:

❍Incremental integration testing-

- Top – Down
- Bottom – up
- Sandwitch/hybrid/Bidirectional

❍Non-incremental integration testing

- Big bang

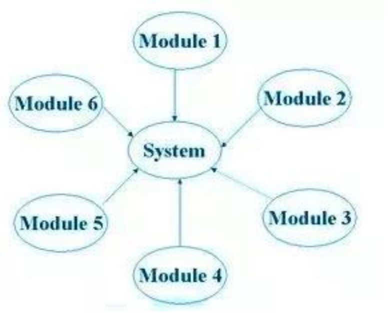
Big Bang Method

In this approach, testing is done only after integration of all modules at once.

It is convenient for small software systems, if used for large software systems identification of defects is difficult.

Since this testing can be done after completion of all modules due to that testing team get for their activities.

So that some modules,requirements,interfaces and high-risk critical modules can be missed easily.



Advantages:

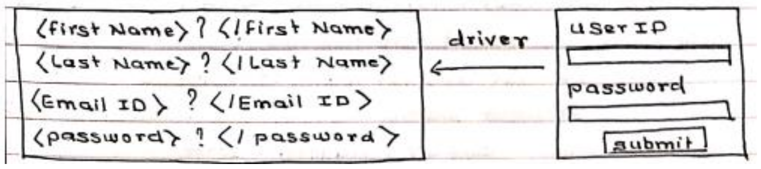
- It is convenient for small size software systems.

Disadvantages:

- Small modules missed easily because of less time for testing
- Time provided for testing is very less.
- We may miss to test some of the interfaces because of less time for testing

Bottom-up Integration Testing

-Bottom-up Integration Testing is a strategy in which the lower level modules are tested first means sub modules tested first.
-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.
-If we have sub module but do not have main module then in that case we use bottom up approach
-To check sub module, developer create dummy main module.
-Developer first create program called “Driver”
-These driver programs are in XML languages.
Example-



Top-down Integration Testing

-If we have to do the integration testing and we have developed module but don't have next module from which we can check correctness of new module.

-Then in this case we use dummy module.

-Dummy module is created from stub.

-Stub is dummy program created by developer

-Stub is in XML format

-Stub use in WSDL(Web service description language)

-We check request and response in an XML

-When we have main module but do not have sub module then we use top down.

-Top Down Integration Testing is a method in which integration testing takes place from top to bottom of software system.

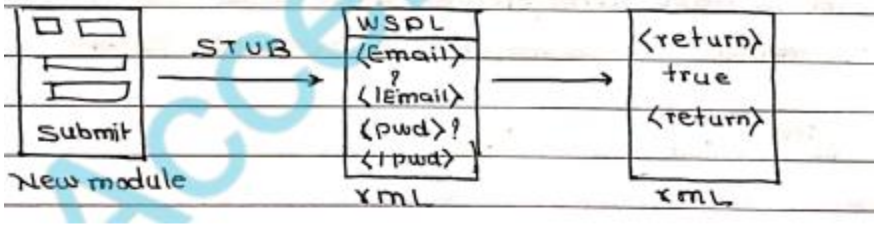
-The higher level modules(main module) are tested first and then lower level modules(sub module) are tested and integrated in order to check the software functionality.

-Stubs are used for testing if some modules are not ready.

Example-

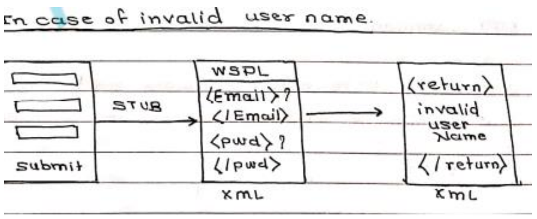
For successful integration-

If both username and password are correct-

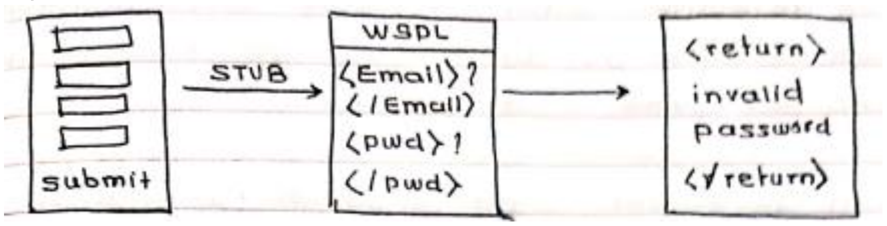


Here we will get the True response/return

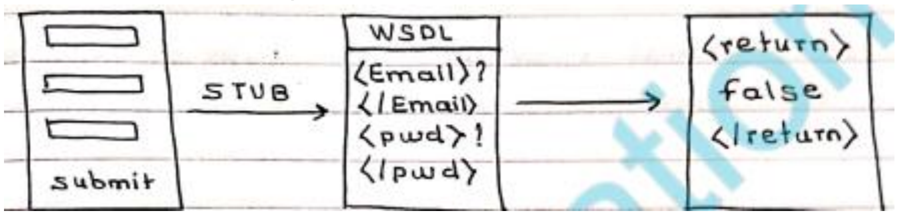
If invalid username-



If password is invalid then-



If both username and password is invalid then-



-Data conversation between client and server done by XML

-XML is the advance version of notepad++

-After updating XML we send the file to server by FTP-file transfer protocol

-Once file sent then we check by response / return from server in database.

-XML data insert in database and same data can collect from database.

Advantages:

- Critical Modules are tested on priority.
- Not completely developed modules/sub modules can be tested without waiting.
- Dependency for any modules can be minimized.

Disadvantages:

- Needs Stubs and drivers

- Developer help needed

<u>sandwich testing/Hybrid testing</u>

-If it is necessary to use both driver as well as stub then the approach is called as Sandwich Approach

-The combination incremental integration testing (I.I.T) and non-incremental integration testing is known as sandwich testing.

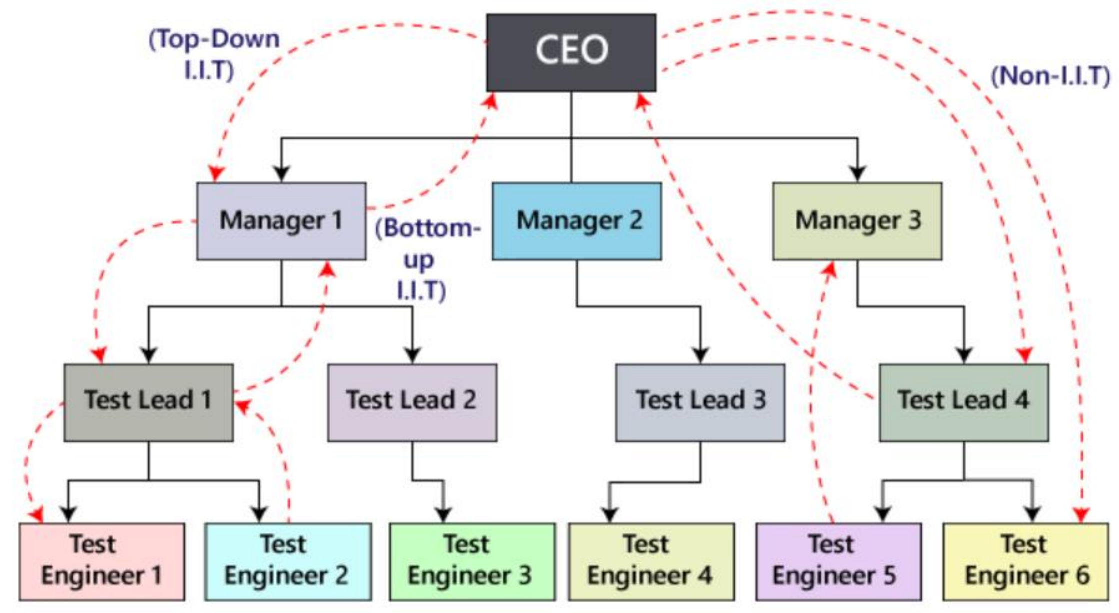
-Here both type of approaches are added and tested.

Example-

In the below example, the development team develops the application and sends it to the CEO of the testing team. Then the CEO will log in to the application and generate the username and password and send a mail to the manager. After that, the CEO will tell them to start testing the application.

Then the manager manages the username and the password and produces a username and password and sends it to the test leads. And the test leads will send it to the test engineers for further testing purposes. This order from the CEO to the test engineer is top-down incremental integrating testing.

In the same way, when the test engineers are done with testing, they send a report to the test leads, who then submit a report to the manager, and the manager will send a report to the CEO. This process is known as Bottom-up incremental integration testing as we can see in the below image:



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
- 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.