Manual testing

What is Software Testing?

- Software Testing is a part of software development process.
- Software Testing is an activity to detect and identify the defects in the software.
- The objective of testing is to release quality product to the client.

Software Quality

- Quality: Quality is defined as justification of all the requirements of a customer in a product.
  - Note: Quality is not defined in the product. It is defined in the customer's mind.
- Quality attributes are varied by
  - Bug free
    Delivered on time
  - Within budget.
  - Meets requirements and/or expectations.
  - Maintainable.

Product Vs Project

- If software application is developed for specific customer based on the requirement then it is called Project.
- If software application is developed for multiple customers based on market requirements then it called Product.

Why do we need testing?

- Ensure that software is bug free.
- Ensure that system meets customer requirements and software specifications.
- Ensure that system meets end user expectations.
- Fixing the bugs identified after release is more expensive.

- Error: Any incorrect human action that produces a problem in the system is called an error.
- Defect/Bug: Deviation from the expected behavior to the actual behavior of the system is called defect.
- Failure: The deviation identified by end-user while using the system is called as failure.

Why the software has bugs normally?

- Miscommunication or no communication
- Software complexity
- Programming errors
- Changing requirements
- Lack of skilled testers

Software Development Life Cycle (SDLC)

- SDLC, Software Development Life Cycle is a process used by software industry to design, develop and test software's.

Waterfall Model(Advantages & Disadvantages)

- Advantages of Waterfall Model
  - Quality of the product will be good.
  - Since Requirement changes are not allowed, chances of finding bugs will be less.
  - Initial investment is less since the testers are hired at the later stages.
  - Preferred for small projects where requirements are fixed.
- Disadvantages of Waterfall Model
  - Requirement changes are not allowed.
  - If there is defect in Requirement that will be carried in later phases.
  - Total investment is more because time taking for rework on defect is time consuming which leads to high investment.
  - Testing will start only after coding.

The goal of a software tester is to find bugs, find them as early as possible, and make sure they get fixed.

WHY TESTING IS NECESSARY?

If any level of testing cannot declare that there is no defect in the product, then Why is it required at all?

Development people assume

that whatever they have developed is as per customer requirements and Will always work. But, it is imperative to create real-life scenario and undertake actual execution of a product at each level of software building (including system level) to assess whether it really works or not.

Developers may have excellent skills of coding but integration issues

can be present when different units do not work together, even though they work independently

One must bring individual units together and make the final product, as some defects may be possible when the sources are developed by people sitting at different places.

The primary role Of software testing is not to demonstrate the correctness of

software product, but to expose hidden defects so that they can be fixed. Testing is done to protect the common users from any failure of system during usage.

Testing is a process Of demonstrating that errors are not present in the product,

This approach is used in acceptance testing where if the application meets acceptance criteria, then it must be accepted by the customer,

Testing gives number of detects present which indirectly gives a measurement of software quality.

More number of defects Vindicate bad software and bad processes Of development,

***software bug* occurs when**

1. The software doesn't do something that the product specification says it should do.
2. The software does something that the product specification says it shouldn't do.
3. The software does something that the product specification doesn't mention

The number one cause of software bugs is the specification

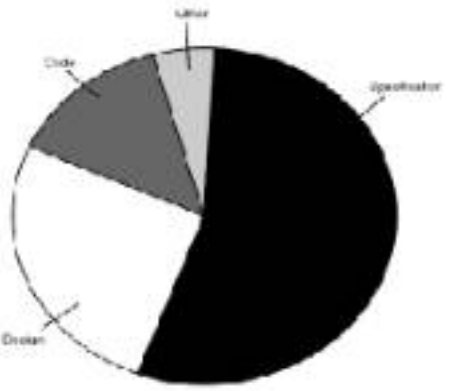


FIGURE 1.1

The bugs are caused for many reasons but main one is Specification.

specifications are the largest bug producer

it's constantly changing, or it's not communicated well to the entire development team.

Planning software is vitally important. If it's not done correctly, bugs will be created.

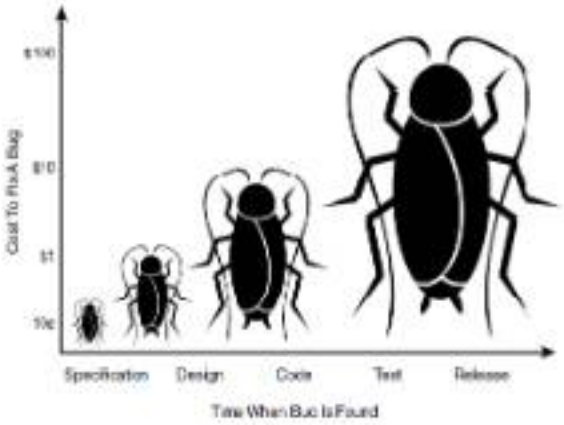
The next largest source of bugs is the design if not designed well then bugs are occurred

Note

There's an old saying, "If you can't say it, you can't do it." This applies perfectly to software development and testing.

This means that wrong design ,, more the bug

The Cost of Bugs



The costs are logarithmic—that is, they increase tenfold as time increases. A bug found and fixed during the early stages when the specification is being written might cost next to nothing, or 10 cents in our example. The same bug, if not found until the software is coded and tested, might cost $1 to $10.

But If a customer finds it, the cost could easily top $100.

Example

root cause of the problem was that the software wouldn't work on a very popular PC platform. If, in the early specification stage, someone had researched what PCs were popular and specified that the software needed to be designed and tested to **work on those configurations**, the **cost** of that effort would have been **almost nothing**.

If that didn't occur, a backup would have been for the software testers to collect samples of the popular PCs and verify the software on them. They would have found the bug, but it would have been more expensive to fix because the software would have to be debugged, fixed, and retested.

The goal of a software tester is to find bugs, find them as early as possible, and make sure they get fixed.

Software testing- It is process to test an application to find out error in it.

Checking the software is ok.

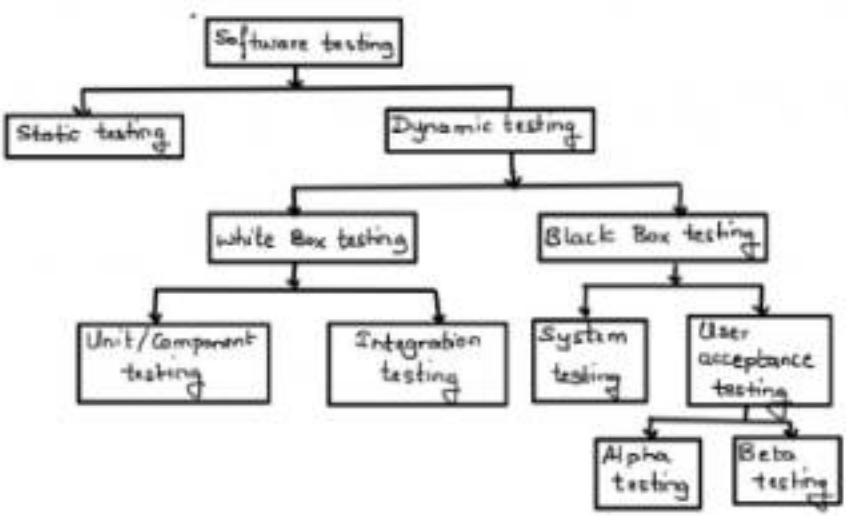
The goal of software tester to find bug.

verifying and validating that a software or application is bug free

Testing Types

1. **Manual testing** :- Manual testing includes testing a software manually, i.e., without using any automated tool or any script

2. **Automation Testing**: Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses another software to test the product. This process involves automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.



Static Testing Techniques

Analysis of a program carried out without executing the program

Done during verification process i.e before development

As we know 85% errors found in desgin phase

Is code is tested in static testing ? "No" || the doocumentation is tested

Software development starts, continues, and ends with documentation

Early documentation → is used to defi ne the software to be built

Later documentation covers → the software training, installation, and operation (user guides)..

Static = not while running

The primary goal of static testing is reduce defect by reducing defects in the

documentation from which the software is developed

Review : is type of static testing | done before execution

Review is a process or meeting during which a work product or set of work products, is presented to managers, users, customers, or other interested parties for comment or

walkthroug Review :

- It is not a formal process
- It is led by the authors
- Author guide the participants through the document according to his or her thought process to achieve a common understanding and to gather feedback.
- Useful for the people if they are not from the software discipline, who are not used to or cannot easily understand software development process.

Insepction Review:

It is the most formal review type

It is led by the trained moderators

During inspection the documents are prepared and checked thoroughly by the reviewers before the meeting

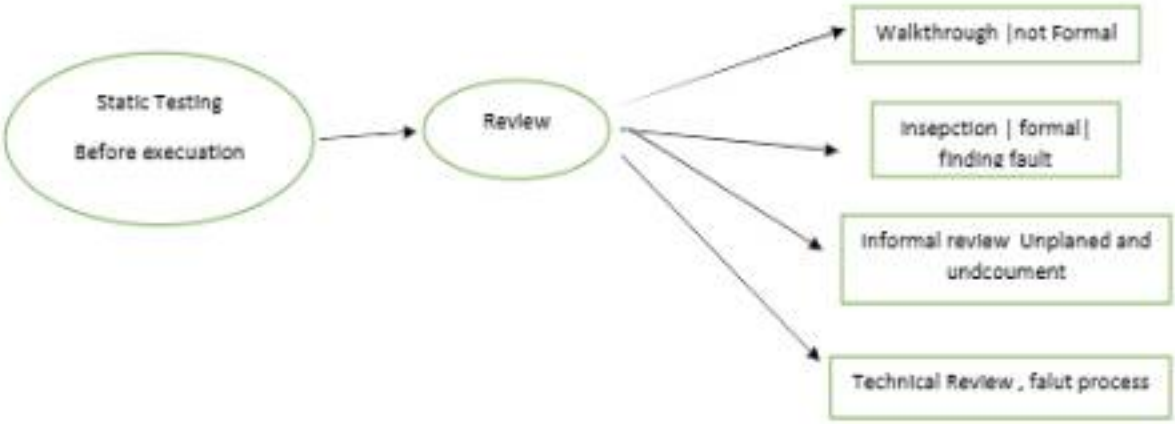
Informal Review

Unplanned and Undocumented

Technical Review

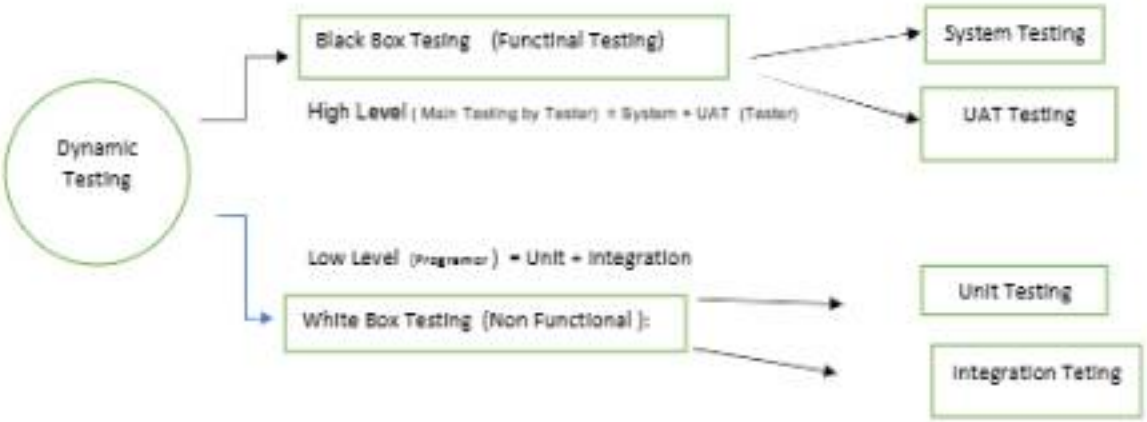
Documented || Defined fault detection process || Includes peers and technical experts

No management participant



Dynamic Testing Techniques

The process of evaluating a system or component based upon its behaviour during execution.



Levels Of Testing

1. Unit Testing :- In unit testing individual component of software tested. The purpose of this testing is that each module is working properly
   It focuses on the smallest unit of software design
   (done by developer by using sample input and observing its sample output)

   Eg. In a program we are checking if loop, method or

   function is working fine

2. Integration Testing: In Integration Testing individual units are combined and tested as group (developer)
   (i) Top-down
   (ii) Bottom-up
   (iii) Sandwich
   (iv) Big-Bang

   Main Purpose of Integration Testing :

   To check moudules are communicating each other as DFD Data Flow Diagram which is specified in TDD (Technical Document Diagram)

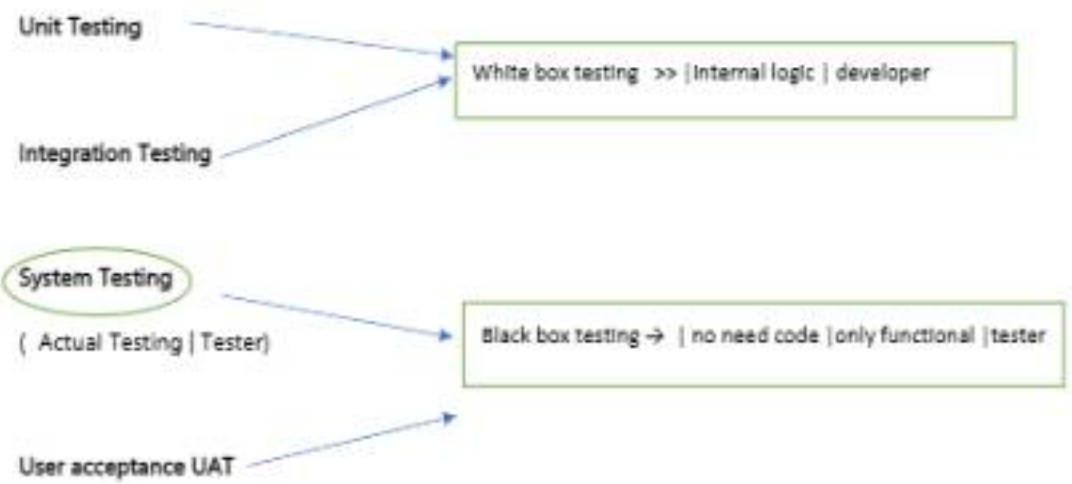
3. System Testing : In this testing we can test whole application (complete / Integrated software is tested) done by tester

4. Acceptance Testing : a level of software Testing in which software is tested for user acceptance

   UAT done at client location where software is actually used

   → Alpha Testing : done by tester in company in presence of customer

   → Beta Testing : done by customer to check software is ok , satisfy requirement



Testing Types

Functional testing

is the process through which QAs determine if a piece of software is acting in accordance with pre-determined requirements. It uses black-box testing techniques, in which the tester has no knowledge of the internal system logic. **Functional testing is only concerned with validating if a system works as intende**

Functional Testing : Testing what system does

Functional testing is a type of black-box testing. *"does this actually work?"*
The ultimate goal of functional testing is to ensure that software works according to specifications and user expectations

input values
test cases
Compare actual and expected output

Eg. Login funcatinallity, registration funcationality

1. **Non Functional Testing :**

- Load testing
- Reliability
- The readiness of a system
- Usability testing

Eg. A practical example would be: checking how many people can simultaneously check out of a shopping basket

2. **Black Box Testing : (without code) High level**

Black box testing is that kind of software testing you can do when you do not have the source code, just the executable code in hand

The testing is done without the internal knowledge of the products

3. **White box testing : (with code) low level**

Monitoring internal structure . check internal logic . done by developer

4. **Smoke testing** : - (Testing on newly released build → **compulsary requirement**)

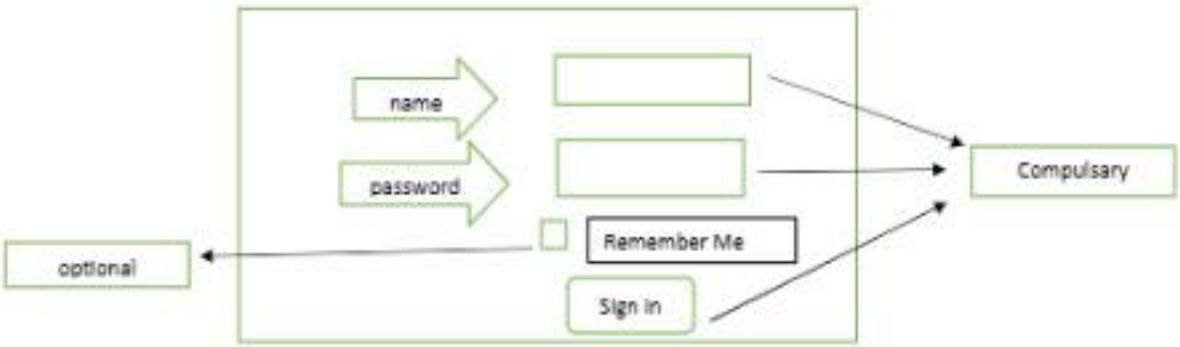
It is first testing on newly released build ... (Build Verification Testing)

Check → the deployed software build is stable or not.

Eg

QC → Build s/w → QA → Testing

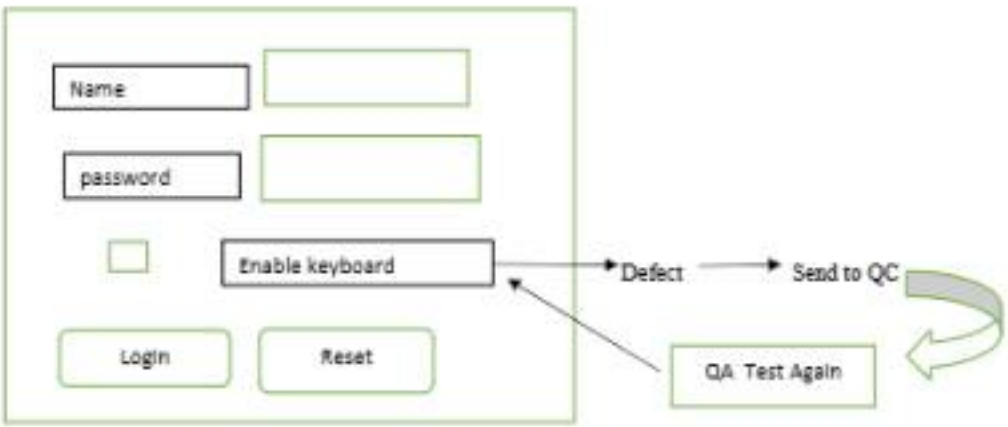
Compulsory Testing → smoke testing



Sanity Testing : (Testing on newly released build → check **Compulsory + optional**)

In above fig all filed are compulary then it is sanity Testing

Retesting : Testing functionality once again



Regression Testing
(Re- running if code changes) :

It is overall testing when ever new change is occurred .

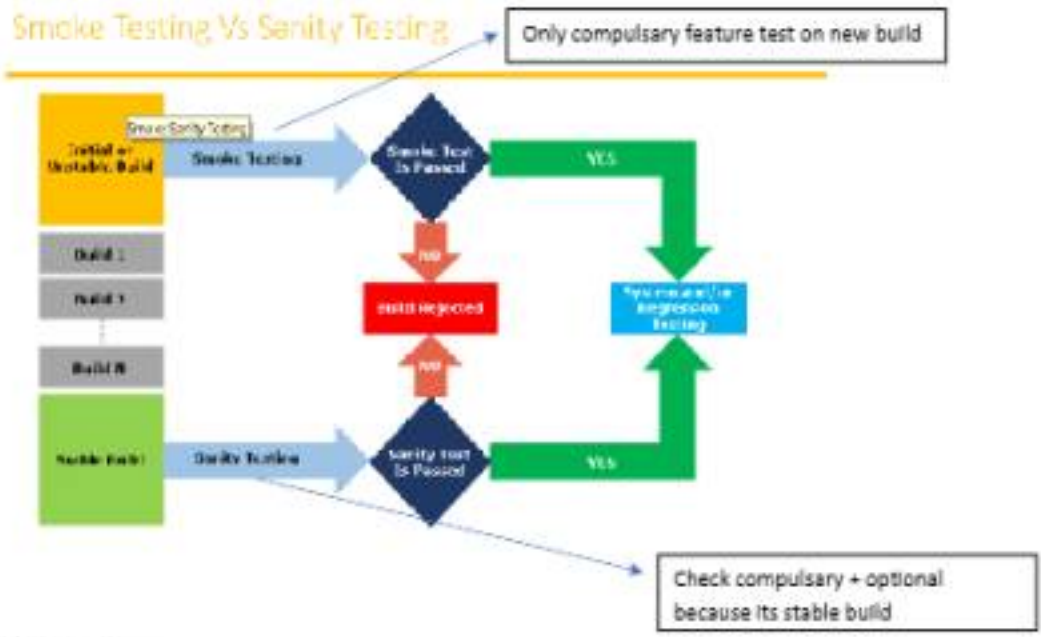
Re-running functional + non Functional test

Code change , does not impact existing function.

After change code , is software works Ok.

Every time a new module is added leads to changes in the program.

This type of testing makes sure that the whole component works properly even after adding components to the complete program.



Static Testing :

Review
Inspection
Walkthrough

Test software without executing software (just test documents)

This test done for avoid bugs in early stage (review testing) look and feel

Why Static testing

→ Early defect detection and correction
→ To get fewer defect at a later stage of testing

Static technique has 3 types

Review : review before development i.e simple document

Read the document, correct or not
Document s/b correct and complete
Requirement review , design review , Test plan review
Review testing can do anybody , manager|developer|tester| coworker etc.

Walkthorough :

Its is informal , anytime , not planned , done when ever required
author of document will explain to their team

Inspection :

Most formal , 3- 8 people in meeting
Proper meeting , schedule which is intimated by mail

Dynamic Testing : **A testing which is done after code development.**

The main purpose of dynamic testing is to test software behaviour with dynamic variables

dynamic testing requires code to be executed

static testing → just analyse the code, no need execution

Alpha Tesing : its is final testing in development

Advantage : immediate solution is possible

Beta Testing : it is 1st testing in client side . it is also called user user acceptance testing UAT

Disadvantage : no immediate solution if defect is found

Installation Testing : providing required resources at client location

It is type of testing in which test engineer check deployment process is successful as per user guideline

Deployment document /user manual : it is document prepared by project manager

Usability Testing : checking application for user friendliness

Monkey Testing : used for game testing, used for random input

To check the application or system will crash

Portablity Testing : Developed application Shoud support multiple enviorment

Forced error Testing : to check valid error message will display

Exploratary Testing : When test enginer does not have idea of functinal testing then he is learnig through exploring application

End to End Testing : We can check all internal componant for successful response

Internal componant like Client , Network, Server Database etc are working fine

Means Testing internal componant

Security Testing : Checking Seculrity of application

Relablity Testing : The Developed application Should Support Longer Duration i.e. Stability

Audit : It is independent evolution of software .

Inspection : It is formal evolution of software

Concurrency Testing : multiuser Testing

Debugging : executing program line by line for finding errors.

Some of the most popular SDLC models are:

→ *no matter which module is used each has same phases*

* Waterfall Model
* V-Shaped Model
* Incremental Life Cycle Model
* Spiral Model

SDLC : software development life cycle

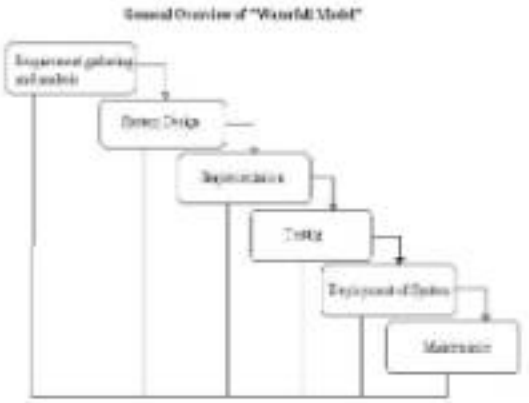
It is process used by software company to develop , design , test software .

1. Requirement analysis
2. Design (blue print)
3. Coding or development
4. Testing
5. Maintenance

Waterfall Module : It is old and traditional model

It is linear model i.e. steps 1 after another

Each phase must complete to start new phase (i.e called one after another)



→ In waterfall module quality of product is good because every phase has clear Documentation

→ SRS (system requirement specification) not changed hence no bug

→ Initial Investment is less because no tester involved

→ no changes in middle

→ testing will start after coding

V shaped Model : verification and validation

Verification → done before development – check we are doing correct ?

Verifying document >> because no software ready

Verification = before s/w = static

Review | walkthrough | inspection

Static Testing = verifying doc

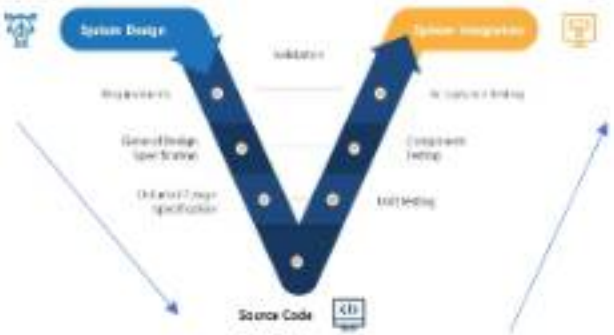
Validation – Actually testing software – done after software ready – done Right ?

Product is ready just check ok or not

Validation = after s/w = dynamic

Dynamic Testing = unit | integration | system | user acceptance

V-Cycle PowerPoint Diagram



System design = before development = verification = static testing = verify documents

System Integration =after devep = validation = dynamic testing = unit | integration | system | UAT

In V model = testing is involved every phase

Disadvantage : more documents

Spiral Model

Spiral Model

- Spiral Model is iterative model.
- Spiral Model overcome drawbacks of Waterfall model.
- We follow spiral model wherever there is dependency on the modules.
- In every cycle new software will be released to customer.
- Software will be released in multiple versions. So it is also called version control model.

Spiral Model (Advantages & Disadvantages)

- Advantages of Spiral Model
  - Testing is done in every cycle, before going to the next cycle.
  - Customer will get to use the software for every module.
  - Requirement changes are allowed after every cycle before going to the next cycle.
- Disadvantages of Spiral Model
  - Requirement changes are NOT allowed in between the cycle.
  - Every cycle of spiral model looks like waterfall model.
  - There is no testing in requirement & design phase.

Incremental Model : requirement are divided into multiple module each module goes through SDLC phase i.e. analysis, desing, coding , testing , maintenance

Requirement → module 1 + module 2 +..... module n

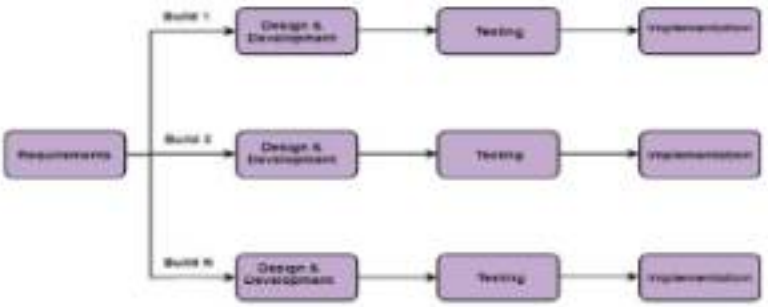


Fig: Incremental Model

When use incremental model :

→ A project has a lengthy development schedule.

→ When the requirements are superior.

Disadvantages :

- Need for good planning
- Total Cost is high.

QA | QC | QE

QA = Highest Possible quality

quality assurance | process related | high level management

Process designed by QA | Responsible for highest possible quality

QC = quality control | product related | actual tester

QA is process orientated | QC product orientated (that work on product actual tester)

QA = responsible for preventing defect (involves all phase i.e. design coding ..)

QC = responsible for finding defect (involves only testing)

QE = Quality engineer

Responsible for write code for testing (automation engineer)

System Testing

(Actual Testing)

System Testing : GUI Testing | Usability Testing | Functional Testing | non functional

1. GUI Testing :
   - :- Testing GUI application , user interface testing
   - :- such as menus , check boxes , icon, images
   - :- not functional , just look and feel
   - :- check size and position of element
   - :- image quality, spelling check , alignment
   - :- Fonts are understanding or not

2. Usability Testing :
   - :- check the easiness of application
   - :- helping messages are display if user confuse
   - :- check user friendly application or not ?

3. Functional Testing :

:- check behaviour of application

:- check database testing (work with database ok ?)

:- error handling , display error message ok ?

:- calculation and manipulation

Eg. 5+5 = 15 (user requirement) => we follow this

5+5 = 10 (math calculation)

:- check text box disable or enable as user requirement

:- Check database operation DML table, column , records etc

Checking database operation

Black box testing + white box testing = Gray box Testing

4. Non Functional testing :

:- once functional testing done i.e. s/w work user requirement then do non functional testing

:- performance testing

Load testing – gradually increase the load

Stress testing – suddenly increase the load (Eg. Online filling form)

Volume testing – how much data handle

:- security of software

:- recovery of application

:- Compatibility testing – work with all platform

End to End Testing:

:- testing overall application after including all module

Eg : login → add customer

→ delete and edit customer

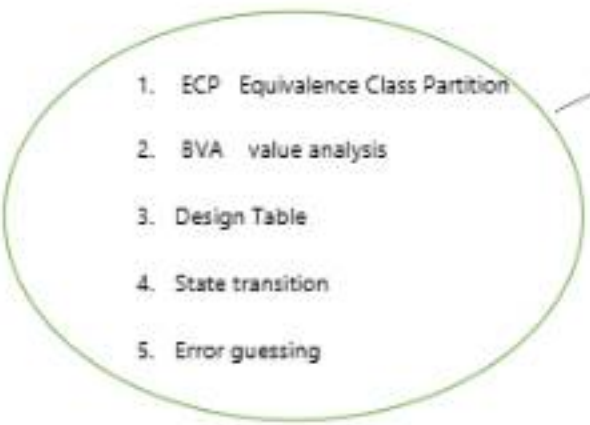
→ logout

Testing all function i.e. add delete edit and logout

Test Case Design Technique:

It helps better design and reduce the number of test case to be executed

Reduce data and more coverage



Black Box Testing

(Just design test case & run

All this are Test Case Design technique

(all this are in Black Box , Its Main testing)

ECP → Equivalence Class Partition

:- value check

:- classify | divide | partition data in → multiple classes to save testing time

Eg.

Enter number [ ] * allow digit form 1-500

Normal test data to check txt box value	Divide data in ECP	Final test data after ECP done to test case
1	-100 to 0 → -50 (invalid)	-50
2	1 to 100 → 30 (valid)	30
3	101 to 200 → 160 (valid)	160
4	201 to 300 → 250 (valid)	250
5	301 to 400 → 350 (valid)	350
6	401 to 500 → 420 (valid)	420
Up to 500	501 to 600 → 550 (invalid)	550

Equivalence Class Partition (ECP)

Name: [ ] * Allow only alphabets

Divide values into Equivalence Classes

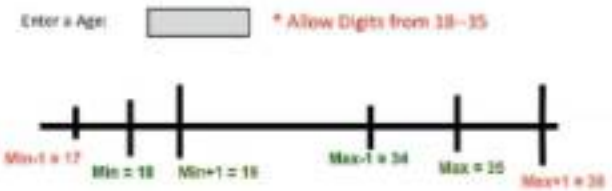
A..Z → (Valid)
a..z → (Valid)
Special Characters → (Invalid)
Spaces → **250** (Invalid)
Numbers → **320** (Invalid)

Test Data using ECP
XYZ
zyz
@#$%
Xy z
1234

BVA: Boundary value Analysis:

Boundary Value Analysis (BVA)

- BVA technique used to check Boundaries of the input.



Min = 18 (Pass)
Min-1 =17 (Fail)
Min+1 =19 (Pass)

Max = 35 (Pass)
Max-1 =34 (Pass)
Max+1 =36 (Fail)

Design table technique:

this technique is used if we have more conditions and based on condition we have to perform the action

eg. transfer money from account to account

Condition :

1. Account no has to approved ✓
2. OTP Matched ✓
3. Sufficient money in account ✓

If condition are ok then do action

Action :

1. Transfer money
2. Show message insufficient money (if any)
3. Block if any suspicious activity

		TC1	TC2	TC3	TC4	TC5
Condition1	Account already approved	TRUE	TRUE	TRUE	TRUE	FALSE
Condition2	OTP Matched	TRUE	TRUE	FALSE	FALSE	X
Condition3	Sufficient Money in the Account	TRUE	FALSE	TRUE	FALSE	X
Action1	Transfer Money	Execute				
Action2	Show message 'Insufficient Amount'		Execute			
Action3	Block the transaction incase of Suspicios Transaction			Execute	Execute	X

State Transition Technique :

Take action depends of on state

- Take an example of login page of an application which locks the user name after three wrong attempts of password.

STATE	LOGIN	CORRECT PASSWORD	INCORRECT PASSWORD
S1	First Attempt	S4	S2
S2	Second Attempt	S4	S3
S3	Third Attempt	S4	S5
S4	Home Page		
S5	Display a message as "Account Locked, please contact Administrator"		

Error Guessing Technique :

:- no any specific rule

:- this test based on tester skill eg. Submit form empty and guess error

Test case Scenario – simply the name of test what to test (name of Test)

Test case – how to test i.e. step

Group of steps that is to be executed to check functionality

Eg. Test scenario = check functionality of login button

Test case = TC01 , TC02 , TC03 , ... etc

Test suite – group of test case

Test case document

:- Test case Id

:- Test case Title

:- Description

: Precondition

:- Priority

:- Request id

:- Steps/ Action

:- Excepted result

:- Actual result

:- Test Data

Project Name:	Google Email
Module Name:	Login
Reference Document:	Diary
Created by:	Rajkumar
Date of creation:	DD-MMM-YY
Date of review:	DD-MMM-YY

STM

TEST CASE ID	TEST SCENARIO	TEST CASE	PRE CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/ FAIL)
TC_LOGIN_001	Verify the ops of Gmail	Enter valid User Name and valid Password	..Need a valid Gmail Account to login	1. User User Name 2. User Password 3. Click 'Login' button	Valid user name valid password	Successful login	Gmail inbox is shown		
TC_LOGIN_002	Verify the login of Gmail	Enter valid User Name and Invalid Password	..Need a valid Gmail Account to login	1. User User Name 2. User Password 3. Click 'Login' button	Valid user name Invalid Password	A message The email or password you entered is not valid/ is close			
TC_LOGIN_003	Verify the ops of Gmail	Enter invalid User Name and valid Password	..Need a valid Gmail Account to login	1. User User Name 2. User Password 3. Click 'Login' button	Invalid user name valid password	A message The email or password you entered is not valid/ is close			
TC_LOGIN_004	Verify the login of Gmail	Enter invalid User Name and Invalid Password	..Need a valid Gmail Account to login	1. User User Name 2. User Password 3. Click 'Login' button	Invalid user name Invalid Password	A message The email or password you entered is not valid/ is close			

Requirement Traceblity matrix : (RTM)

:- Trace how many Test case are executed or covered

: In simple keep track of test cases

Test Metrics

SNO	Required Data
1	No. Of Requirements
2	Avg. No. of Test Cases written Per Requirement
3	Total No.of Test Cases written for all Requirement
4	Total No. Of test cases Executed
5	No.of Test Cases Passed
6	No.of Test Cases Failed
7	No.of Test cases Blocked
8	No. Of Test Cases Un Executed
9	Total No. Of Defects Identified
10	Critical Defects Count
11	Higher Defects Count
12	Medium Defects Count
13	Low Defects Count
14	Customer Defects
15	No.of defects found in UAT

Test Case Execution:

Executing test case based on test plan

Mark status Pass | Fail | Blocked

Reports defects in bug report

Defect Reporting Tool :

ClearQuest :- only bug report

Devtrack : only bug report

Jira – test management tool (track each activity)

Bugzilla – test management tool (track each activity)

Defect Report Details :

:- Defect id

:- Defect Version

:- Step : details of step along with developer what to do

:- date

: - detected by

:- status

:- fixed by in process | fixed

: severity – impact

:- priority – high| medium | low

Severity Of defect :- Blocker | Critical | Major | Minor

Seriousness of application

Testing engineer decides the severity level of the defect.

Blocker : this defect show application not processed

Critical : main function not working

Major : some undesirable behaviour eg. Email sent but msg not display

Minor : look and feel

Priority of defect: - High | Medium | Low

Importance of defect

On which priority defect will be solved or fixed

P0 - High fixed immediately in same version

P1 - Medium fixed in next release

P2- Low next version

Manual Testing project: E commerce

:- Project introduction

:- Understanding and explore the functionality

:- Test Plan

:- Writing test scenario

:- Writing test cases

:- Environment setup and build and development

:- Test execution

:- Bug reporting and tracking

:- Sanity testing , smoke testing , regression testing

:- test sign off

E commerce project

- Login
- Search for product and item
- Add them to cart
- Do payment
- Product will be delivered
- Return the product
- Etc.

SRS document

A **software requirements specification** (SRS) is a document that describes

what the software will do and how it will be expected to perform.

Refer SRS Document

E commerce project

:- Project Information

:- Understanding the functionality of project

:- Test Plan → a detailed document of testing activity

:- writing test scenario

:- Test case and review

:- Environment setup and deployment for testing application

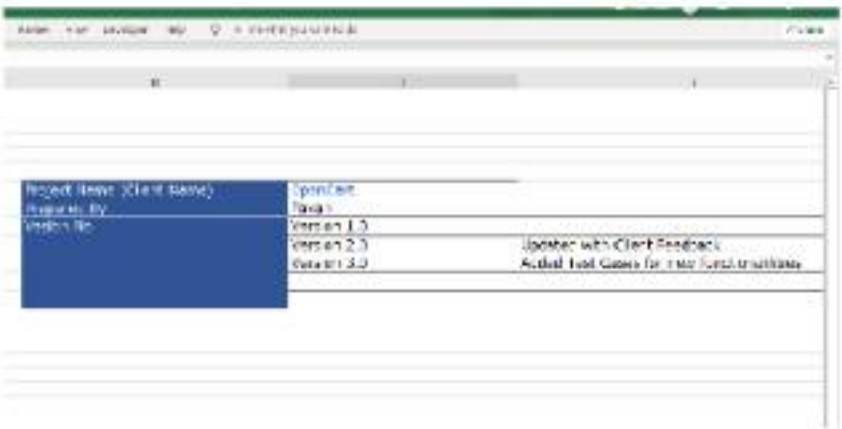
:- **Test Execution**

:- bug reporting and tracking

:- Sanity and regression testing

:- Test Sign off

Version page



Opencart.com project

1. FRS Document : → How Software works

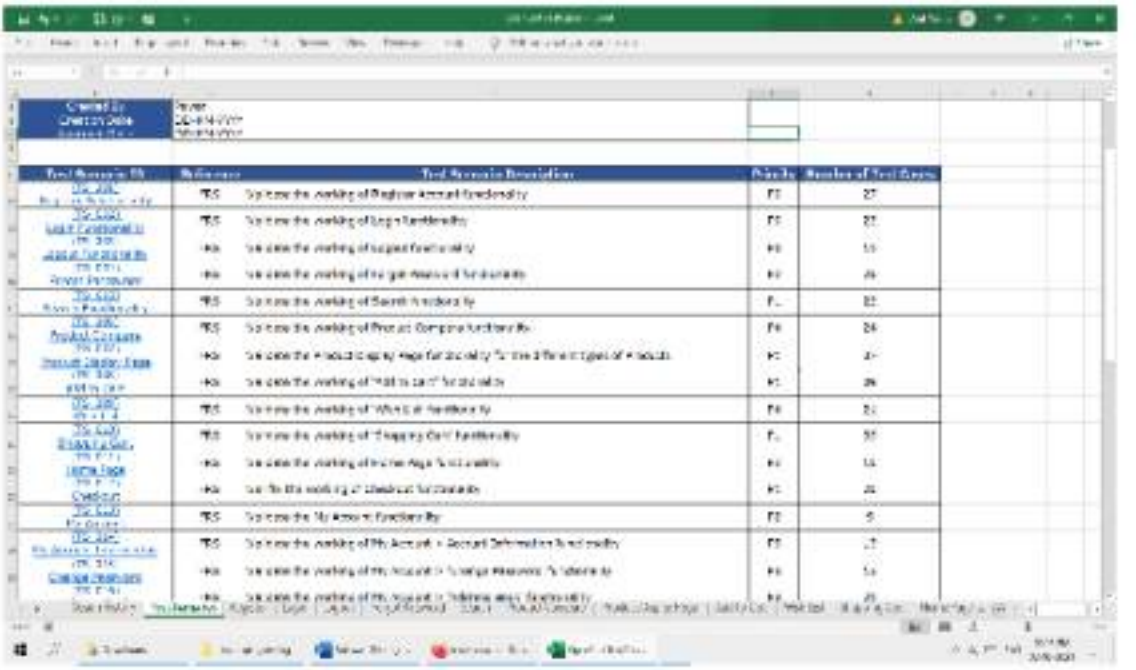
   An FRS, or **functional requirement specification** is the document that describes all the functions that software or product has to perform

2 Test Plan Document → a detailed document of testing activity

3. Test Scenario Document : → Anything that can be tested is a *Test Scenario*.

   → Simply name of test case

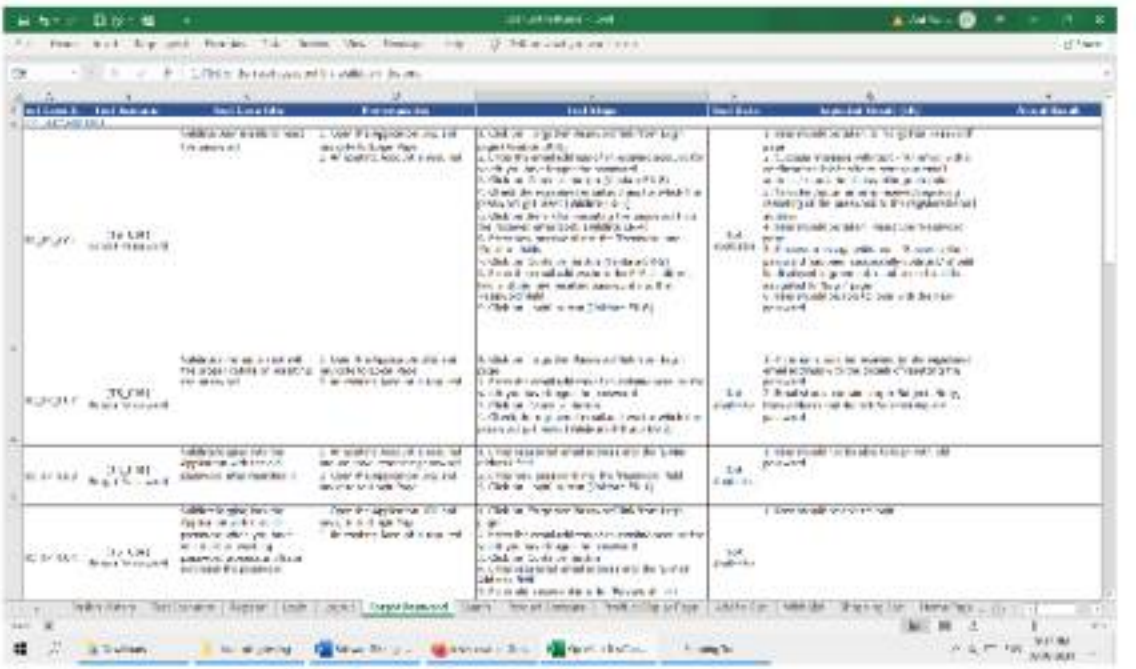
   Test Scenario Document



4. Test Case Design :

While writing test case refer the test Test Scenario and FRS Document

Test case template



All above are sample test case

After executing above steps i.e. test case update the field

Actual result

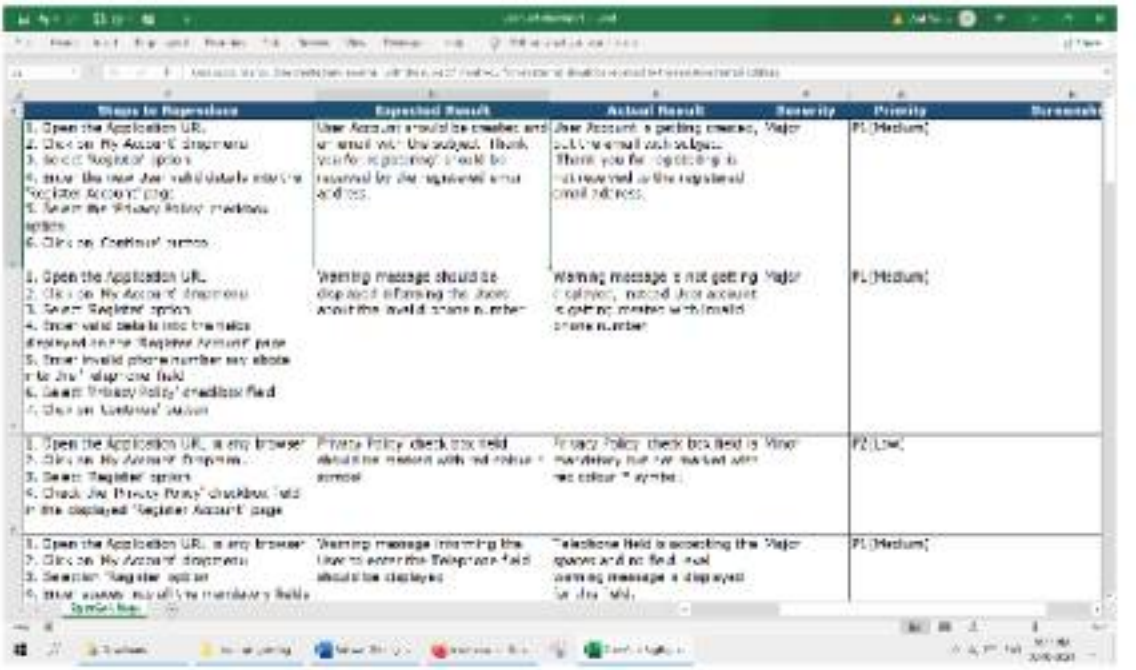
Result – pass or fail

Priority

After executing test case maintain or update sheet RTM eg. No test executing or blocked etc

If the bug found in testing report it in bug tracking file

Eg.



Steps to Reproduce	Expected Result	Actual Result	Severity	Priority	Screenshot
1. Open the Application URL 2. Click on My Account' dropmenu 3. Select 'Register' option 4. Enter the new user valid details into the 'Register Account' page 5. Select the 'Privacy Policy' checkbox option 6. Click on 'Continue' button	User Account should be created and an email with the subject 'Thank you for registering' should be received by the registered email address.	User Account is getting created, Major bug. We've email with subject 'Thank you for registering' is not received at the registered email address.		P1 [Medium]	
1. Open the Application URL 2. Click on 'My Account' dropmenu 3. Select 'Register' option 4. Enter valid details into the fields displayed on the 'Register Account' page 5. Enter invalid phone number say abcde in the 'Telephone' field 6. Select 'Privacy Policy' checkbox field 7. Click on 'Continue' button	Warning message should be displayed informing the 'Enter valid the invalid phone number'	Warning message is not getting displayed, instead User account is getting created with invalid phone number	Major	P1 [Medium]	
1. Open the Application URL in any browser 2. Click on 'My Account' dropmenu 3. Select 'Register' option 4. Check the 'Private Policy' checkbox field in the displayed 'Register Account' page	Private Policy check box field should be marked with red colour symbol	Privacy Policy check box field is 'Yes' mandatory but 'Yes' marked with red colour '*' symbol		P2 [Low]	
1. Open the Application URL in any browser 2. Click on 'My Account' dropmenu 3. Select 'Register' option 4. Enter values in all the mandatory fields	Warning message (reminding the User to enter the Telephone field should be displayed	Telephone field is accepting the spaces and no field value warning message is displayed for this field.	Major	P1 [Medium]	

Test case with result

Big Bang Testing Approach

Big bang' approach involve testing software system after development work is completed. This is also termed *system testing' or final testing done before releasing software to the customer for acceptance testing.

Big Bang == System Testing== Final Testing == Before Release

This is last part Of Software development as per waterfall methodlogy

6.2.Testing Vs Debugging

Testing	Debugging
Testing is done: to find bugs	Debugging is an art of fixing bugs

- **Black Box testing:** Mainly perform by testers
- **White box testing:** Mainly perform by developers
- **Unit testing:** Part of White box testing
- **Acceptance testing:** This is the final testing done by Customer based on the agreements Load / stress / performance testing : Testing an application load capacity
- **Usability testing:** Testing to determine the user friendly ness of the application
- **Install / Uninstall testing:** Testing of full, partial, or upgrade install / uninstall processes.
- **Recovery / failover testing:** Testing to determine how well a system recovers from crashes, failures, or other major problems.
- **Incremental integration testing:** Continuous testing of an application as new functionality is added
- **Ad-hoc testing:** Conducting testing without requirements
- **Comparison testing:** Comparing software weaknesses and strengths to competing products.
- **Alpha testing:** Part of UAT
- **Beta testing:** Part of UAT
- **Integration testing:** validating combined modules of an application
- **Functional testing:** part of black box testing
- **System testing:** part of black box testing and validating the system requirements
- **End to End testing:** similar to system testing
- **Sanity testing or smoke testing:** An initial validation of a New build or release
- **Regression testing:** validating the existing functionality of the application once new fixes added
- **Compatibility testing:** Testing an application in different environments.

A bug is a issue or error in code or any environmental issue.

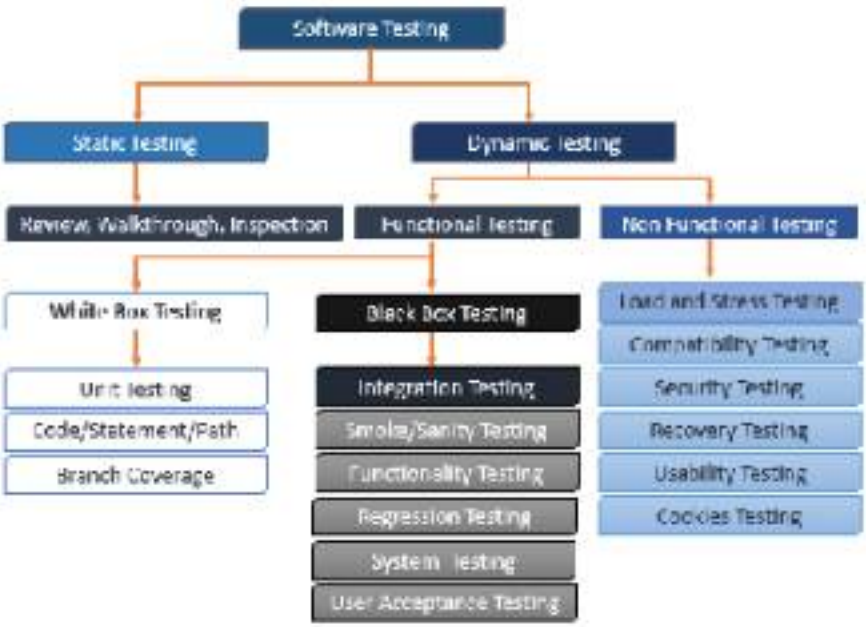
What is Test Case?

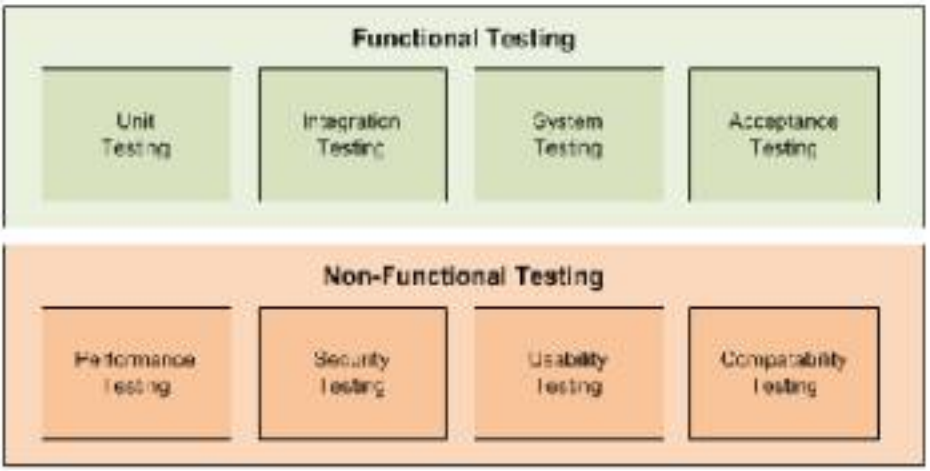
A test case is a detailed explanation of a scenario. Test case is a document which describes pre-condition, post-condition, test data, actors and navigation of the particular functionality.Each and every test case should have unique test name and test ID.

What is Test Plan?

Test Plan is a document that explains what to test, when to test, where to test and when to complete

3- Retesting is only done for failed Test cases while Regression is done for passed test cases





Black Box Testing	White Box Testing
1. Black box testing techniques are also called functional testing techniques	1. White box testing techniques are also called structural testing techniques
2Black Box Testing is a software testing method in which the internal structure/ design/ implementation of the item being tested is NOT known to the tester	2. White Box Testing is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester.
3. It is mainly applicable to higher level of testing such as Acceptance Testing and System Testing	3. Mainly applicable to lower level of testing such as Unit Testing and Integration Testing
4. Black box testing is generally done by Software Testers	4. White box testing is generally done by Software Developers
5. Programming knowledge is not required	5. Programming knowledge is required
6. Implementation knowledge is not required	6. Implementation knowledge is required

	minimum or maximum value of a range.
Boundary value analysis	A black box test design technique in which test cases are designed based on boundary values.
Branch testing	A white box test design technique in which test cases are designed to execute branches.
Business process-based testing	An approach to testing in which test cases are designed based on descriptions and/ or knowledge of business processes.
Capture/ playback tool	A type of test execution tool where inputs are recorded during manual testing in order to generate automated test scripts that can be executed later (i.e. replayed). These tools are often used to support automated regression testing.
Certification	The process of confirming that a component, system or person complies with its specified requirements, e.g. by passing an exam.
Code coverage	An analysis method that determines which parts of the software have been executed (covered) by the test suite and which parts have not been executed, e.g. statement coverage, decision coverage or condition coverage.
Compliance testing	The process of testing to determine the compliance of the component or system.
Component integration testing	Testing performed to expose defects in the interfaces and interaction between integrated components.
Condition testing	A white box test design technique in which test cases are designed to execute condition outcomes.
Conversion testing	Testing of software used to convert data from existing systems for use in replacement systems.
Data driven testing	A scripting technique that stores test input and expected results in a table or spreadsheet, so that a single control script can execute all of the tests in the table. Data driven testing is often used to support the application of test execution tools such as capture/ playback tools.
Database integrity testing	Testing the methods and processes used to access and manage the database, to ensure access methods, processes and data rules function as expected and that during access to the database, data is not corrupted or unexpectedly deleted, updated or created.
Defect	A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system.
Defect masking	An occurrence in which one defect prevents the detection of another.

Acceptance testing	Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.
Ad hoc testing	Testing carried out informally; no formal test preparation takes place, no recognized test design technique is used, there are no expectations for results and arbitrariness guides the test execution activity.
Agile testing	Testing practice for a project using agile methodologies, such as extreme programming (XP), treating development as the customer of testing and emphasizing the test-first design paradigm.
Alpha testing	Simulated or actual operational testing by potential users/ customers or an independent test team at the developers' site, but outside the development organization. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing.
Back-to-back testing	Testing in which two or more variants of a component or system are executed with the same inputs, the outputs compared, and analyzed in cases of discrepancies.
Beta testing	Operational testing by potential and/ or existing users/ customers at an external site not otherwise involved with the developers, to determine whether or not a component or system satisfies the user/ customer needs and fits within the business processes. Beta testing is often employed as a form of external acceptance testing for off-the-shelf software in order to acquire feedback from the market.
Big-bang testing	A type of integration testing in which software elements, hardware elements, or both are combined all at once into a component or an overall system, rather than in stages.
Black-box testing	Testing, either functional or non-functional, without reference to the internal structure of the component or system.
Black-box test design technique	Procedure to derive and/ or select test cases based on an analysis of the specification, either functional or non-functional, of a component or system without reference to its internal structure.
Blocked test case	A test case that cannot be executed because the preconditions for its execution are not fulfilled.
Bottom-up testing	An incremental approach to integration testing where the lowest level components are tested first, and then used to facilitate the testing of higher level components. This process is repeated until the component at the top of the hierarchy is tested.
Boundary value	An input value or output value which is on the edge of an equivalence partition or at the smallest incremental distance on either side of an edge, for example the

Elements of Software Testing

Software testing is an essential service for any business implementing a new system or updating an old one. Software development technologies are constantly changing, but the essential elements for successful software testing remain the same.

GENERAL DEFINITIONS

Qa	Quality Assurance	Monitors the software engineering processes to ensure a high quality product
St	Software Testing	Software testing is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results

SOFTWARE TESTING TYPES

Ut	Unit Testing	Testing units of code using automation to ensure that each unit works individually as it should, and the system as a whole is tested later
Sm	Smoke Testing	A quick and simple test to ensure that the major functions of software work as intended, and typically covers most functionality when new hardware builds are first built
Fu	Functional Testing	Assesses functional components of the system and ensures that it performs the basic functions that it was designed to fulfill
In	Integration Testing	Determines if multiple components work together properly within the system
Sy	System Testing	Analyzes the behavior of the whole system and whether or not it fits its defined requirements
Ua	User Acceptance Testing	Verifies that it can handle real-world scenarios and functions as intended; a requirement that acceptance of the software is performed by actual end users of the system
Li	Live Testing	Tests functions and features that can only be accurately tested on the live system instead of a test environment
Re	Regression Testing	Ensures that changes to the code haven't broken any previously functional parts of the system
Pe	Performance Testing	An umbrella term that includes all processes which evaluate the overall experience of the system
Se	Security Testing	Determines the safety of system data without compromising a system's functionality
Co	Compatibility Testing	Ensures functionality of the system across a variety of browsers and operating systems
Mo	Mobile Testing	Ensures that the application functions across a variety of mobile devices
Ac	Accessibility Testing	Verifies that your software is accessible to customers who have disabilities
Lo	Localization Testing	Ensures that software performs as required in different countries, and that translations are correct
Bb	Black Box Testing	Examines the functionality of an application without knowledge of internal structures
Wb	White Box Testing	Tests internal processes of an application, as opposed to its functionality
St	Static Testing	Examination of the system's code and documentation, without actually using it
Ut	Usability Testing	Technique which evaluates an application on how easy it is for users to use it

TESTING TOOLS

Bt	Bug Tracker	Keeps track of software bugs, usually made up of an automatic flow of bugs, a set of customizable reports, and different system roles are permissions
Tm	Test Management System	Stores testing information from the test planning stage, through execution of cases, up to the reporting stage
Am	Application Lifecycle Management System	Helps a company manage its software development life cycle in a dedicated, centralized place
At	Automation Tools	Control the execution of tests and the comparison of actual outcomes with predicted outcomes
Pt	Performance Testing Tools	Measure an application's response while simulating the traffic of tens or even hundreds of thousands of users performing a wide range of tasks on the application

TESTING STRATEGIES

Rb	Risk-based Testing	Prioritizes test processes based on the risk that flaws pose to an application, and the effect those flaws could have on the company or end user
Ex	Exploratory Testing	Test design and test execution at the same time, using protocol, thoughtful approach and flexible testing
Au	Automated Testing	Uses code scripts to control the execution of tests without manual intervention
Ci	Continuous Integration	Members of a team integrate their work frequently, usually each person integrates at least daily, leading to multiple integrations per day
Rm	Requirements Traceability Matrix	Ways test cases to requirements to ensure testing coverage of requirements

TESTING DOCUMENTATION

Tp	Test Plan	Describes the scope, approach, resources, schedule, test items, features to be tested and not tested, risks, and contingencies for the entire testing process
Tc	Test Case	A set of inputs, preconditions, predicted results and execution conditions for a particular objective or test condition
To	Test Procedure Specifications	A document of actions for the execution of a case
Ts	Test Script	Specifies test procedures, typically for automated testing
Dr	Defect Reports	Reports the flaws/bugs/components or the system itself that caused it/it to perform its function properly
Tu	Test Summary Reports	Summarizes testing activities and results

QUALITEST

Version 1

White Box	Black Box
The code remains visible to the testers.	The code remains hidden from the testers.
It can be called **Structural Testing**.	It can be called **Functional Testing**.
It can be called **Clear Box Testing**.	It can be called **Closed Testing**.
It can be also called low-level testing.	It can be also called high-level testing.
Generally done by the developers.	Generally done by the independent testers.
The detailed design document is required for testing.	Specification Document is required for testing.
Tests the logic and implementation of software.	Tests behavior or functionality of the software. The actual implementation remains hidden from the tester.
Programming knowledge and implementation details are required.	Prior knowledge of programming is not required and implementation details remain hidden from testers.
Types of White Box testing includes Path Testing(Flow Graph and Cyclomatic Complexity), Control Structure Testing(Conditions Testing, Data Flow Testing, Loop Testing).	Types of Black Box testing includes Graph-Based Testing, Equivalence Partitioning, Boundary Value Analysis, Comparison Testing, Orthogonal Array Testing.
Generally, testing tools depend on the programming language.	Generally, testing tools are independent of programming language.

WHITE BOX TESTING

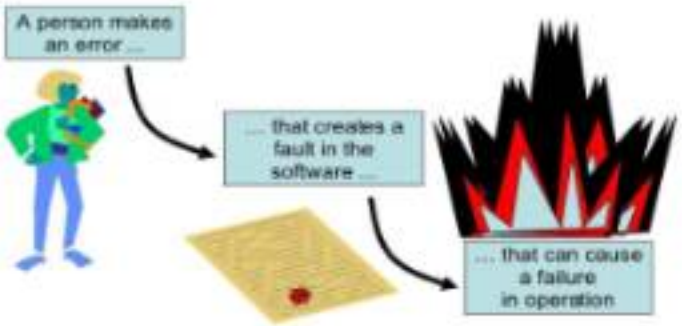
Advantages	Disadvantages
•As the tester has knowledge of the source code, it becomes very easy to find out which type of data can help in testing the application effectively.	•Due to the fact that a skilled tester is needed to perform white box testing, the costs are increased.
•It helps in optimizing the code.	•Sometimes it is impossible to look into every nook and corner to find out hidden errors that may create problems as many paths will go untested.
•Extra lines of code can be removed which can bring in hidden defects.	•It is difficult to maintain white box testing as the use of specialized tools like code analyzers and debugging tools are required.
•Due to the tester's knowledge about the code, maximum coverage is attained during test scenario writing.	

There are two main categories of testing:

1) Static Testing
2) Dynamic Testing

Static Testing	Dynamic Testing
Static testing is completed without executing the program.	Dynamic testing is completed with the execution of program.
This testing is executed in verification stage.	This testing is executed in validation stage.
Static testing is executed before the compilation.	Dynamic testing is executed after the compilation.
This testing prevents the defects.	This testing finds and fixes the defects.
The cost is less for finding the defects and fixes.	The cost is high for finding and fixing the defects.
It consists of Walkthrough, Inspection, reviews etc.	It consists of specification based, structure based, Experience based, unit testing, integration testing etc.

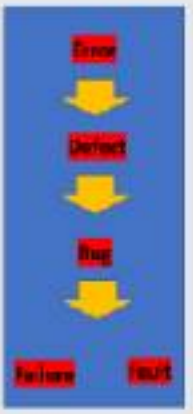
Error - Fault - Failure



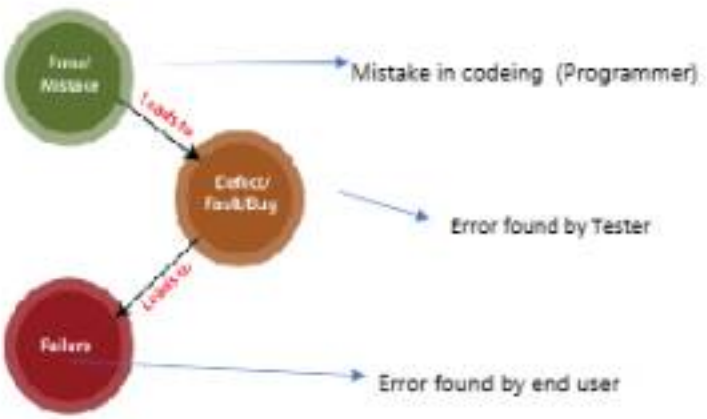
Software Testing

Some terminologies

- Error: A mistake in coding is called Error.
- Defect: Error found by tester is called Defect. The variation between the actual results and expected results is known as defect.
- Bug: Defect accepted by development team then it is called Bug / Anomaly
- Failure: When a defect reaches the end customer it is called a Failure.
- Missing and Wrong: A requirement of the customer that was not fulfilled.
- FAULT: A fault makes an application behave in a wrong manner.



5 Shades of a Defect



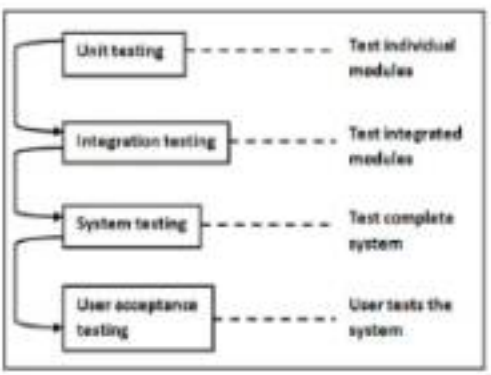
Levels of Testing

Type of Testing	Performed By
• Low-level testing	
– Unit (module) testing	Programmer
– Integration testing	Development team Independent Test Group
• High-level testing	
– System testing	Independent Test Group
– Acceptance testing	Customer/End Users

There are main 4 Lelves of Testing

But these are catagirised by 2 level

High and low level



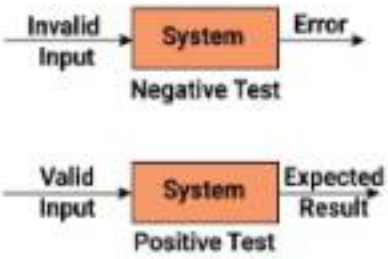
Test Planning : a document that describe how to perform testing on entire application

Quatliyt lead or QA Manager >> Prepare Test Plan

What to test ?

When to test ?

How to test ?



Negative Testing

Age: abcd

Enter only Numbers

BOUNDARY VALUE ANALYSIS (BVA)

AGE [ ] *Accepts value 18 to 56

BOUNDARY VALUE ANALYSIS		
Invalid (min -1)	Valid (min, +min, -max, max)	Invalid (max +1)
17	18, 19, 55, 56	57

Name: [ ] *Accepts characters length (5 - 12)

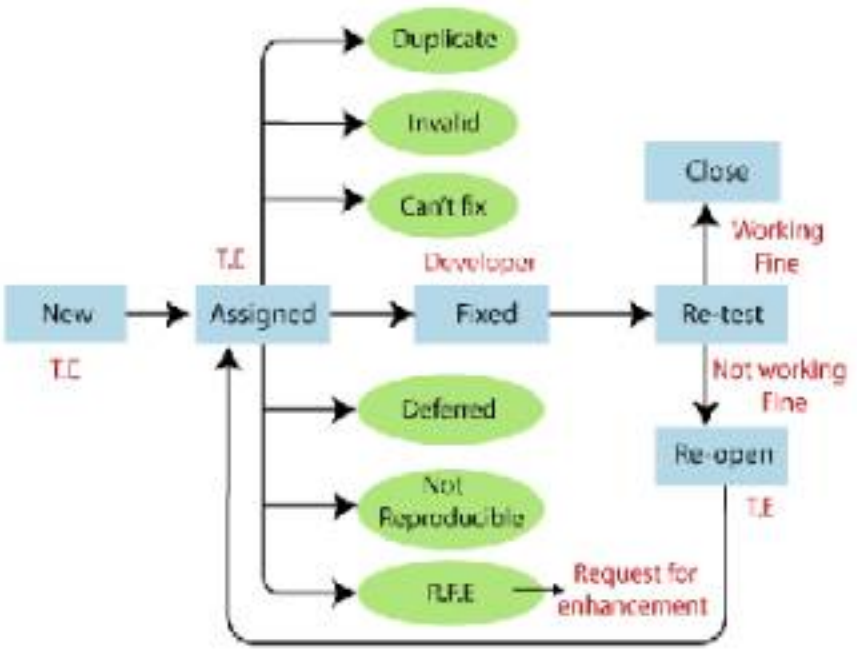
BOUNDARY VALUE ANALYSIS		
Invalid (min -1)	Valid (min, +min, -max, max)	Invalid (max +1)
5 characters	5, 7, 11, 12 characters	13 characters

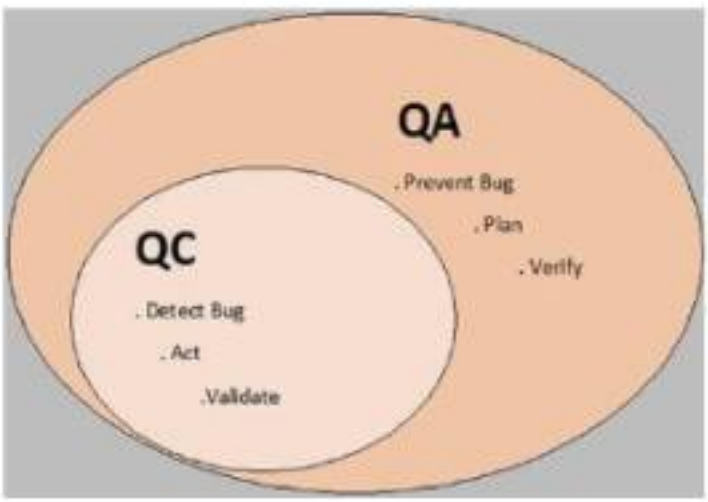
The main use of BVA is to reduce the test cases

It is Black Box Testing because checks functionality

i.e. AUT = Application Under Test

Final Diagram of Bug life cycle





Verification is the process confirming that something—software—**meets its specification**

Validation is the process confirming that it meets the **user's requirements**

Verification = meets specication as document

Validation = meets User requirement (because software is ready hence main tested for User)

Verification : Its static process of analysing the document , not actural product

Validation : it involes Dynamic Testing (unit, integration ,system testing)

Functional Testing	Non- Functional Testing
1. In functional Testing tester finds how well the system performs. 2. Functional Testing is based on client requirements.	1. In Non-Functional Testing tester tests how well the system responds. 2. Non- Functional Testing is based on client expectations.
3. Functional Testing means Testing the application against business requirements.	3. Non Functional Testing means Testing the application against clients and performance requirements
4. It is a part of System Testing. 5. Functional Testing Validating the behavior of application.	4. It is also a part of System Testing 5. Non- Functional Testing Validating the performance of application
6. This Testing covers Unit Testing, Integration Testing, Smoke Testing, Sanity Testing, Regression Testing and so on.	6. This Testing covers Load/Performance Testing, Stress/volume Testing, Security Testing, Installation Testing and so on.
7. It is always concentrating on customer requirements. 8. Functional Testing means how is your system is doing	7. It is always concentrating on customer expectations. 8. Non- Functional Testing means how well your system is doing example usability, performance and stress testing.

Smoke Testing	Sanity Testing
Smoke Test is done to make sure the build we received from the development team is testable/stable or not	Sanity Test is done during the release phase to check for the main functionalities of the application without going deeper.
Smoke Testing is performed by both Developers and Testers	Sanity Testing is performed by Testers alone
Smoke Testing, build may be either stable or unstable	Sanity Testing, build is relatively stable
It is done on initial builds.	It is done on stable builds.
It is a part of basic testing.	It is a part of regression testing.
Usually it is done every time there is a new build releases.	It is planned when there is no enough time to do in-depth testing.

Exploratory Testing

- We have to explore the application ,understand completely and test it.
- Understand the application , identify all possible scenarios , document it then use it for testing
- We do exploratory testing when the Application ready but there is no requirement.
- Test Engineer will do exploratory testing when there is no requirement.
- Drawbacks :
- You might misunderstand any feature as a bug (or) any bug as a feature since you do not have requirement.
- Time consuming
- If there is any bug in application , you will never know about it

Adhoc Testing

- Testing application randomly without any test cases or any business requirement document.
- Adhoc testing is an informal testing type with an aim to break the system.
- Tester should have knowledge of application even though he doesn't have requirements/test cases.
- This testing is usually an unplanned activity.

Monkey/Gorilla Testing

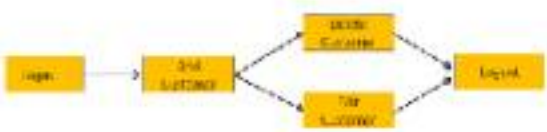
- Testing application randomly without any test cases or any business requirement document.
- Adhoc testing is an informal testing type with an aim to break the system.
- Tester do not have knowledge of application
- Suitable for gaming applications.

Adhoc Testing Vs Monkey Testing Vs Exploratory Testing

Adhoc testing	Monkey Testing	Exploratory Testing
No Documentation	No Documentation	No Documentation
No Plan	No Plan	No Plan
Informal testing	Informal testing	Informal testing
Tester should know Application Functionality	Testers doesn't know Application functionality	Testers doesn't know Application functionality
Random Testing	Random Testing	Random Testing
Intension is to break the application/find out corner defects	Intension is to break the application/find out corner defects	Intension is to learn or explore functionality of application
Any Applications	Gaming Applications	Any Applications which is new to tester

END-to-END testing

- Testing the overall functionalities of the system including the data integration among all the modules is called end-to-end testing.



end-to-end test

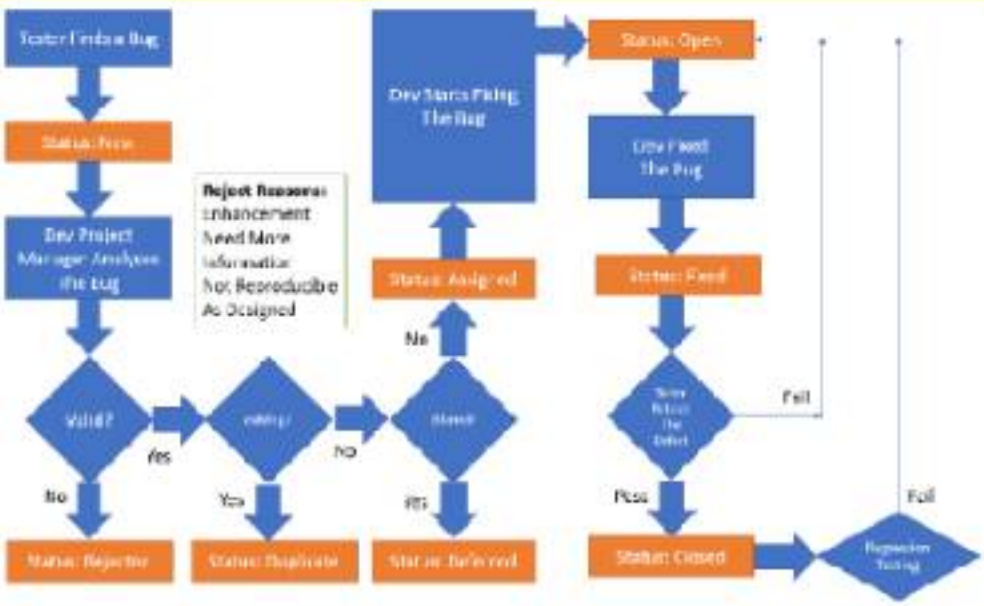
1) Login
2) Add New Customer
3) Edit customer
4) Delete Customer
5) Logout

Globalization and Localization Testing

- **Globalization Testing:**
- Performed to ensure the system or software application can run in any cultural or local environment.
- Different aspects of the software application are tested to ensure that it supports every language and different attributes.
- It tests the different currency formats, mobile number formats and address formats are supported by the applications.
- For example, Facebook.com supports many of the languages and it can be accessed by people of different countries. Hence it is a globalized product.

- **Localization Testing:**
- Performed to check system or software application for a specific geographical and cultural environment.
- Localized product only supports the specific kind of language and is usable only in specific region.
- It tests the specific currency format, mobile number format and address format is working properly or not.
- For example, Baidu.com supports only the Chinese language and can be accessed only by people of few countries. Hence it is a localized product.

Bug Life Cycle



Black Box Testing	White Box Testing
It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.	It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.
It is mostly done by software testers.	It is mostly done by software developers.
No knowledge of implementation is needed.	Knowledge of implementation is required.
It can be referred as outer or external software testing.	It is the inner or the internal software testing.
It is functional test of the software.(Black box)	It is structural test of the software.
This testing can be initiated on the basis of requirement specifications document.	This type of testing of software is started after detail design document.
No knowledge of programming is required.	It is mandatory to have knowledge of programming.
It is the behavior testing of the software.	It is the logic testing of the software.
It is applicable to the higher levels of testing of software.	It is generally applicable to the lower levels of software testing.
It is also called closed testing.	It is also called as clear box testing.
It is least time consuming.	It is most time consuming.
It is not suitable or preferred for algorithm testing.	It is suitable for algorithm testing.
Can be done by trial and error ways and methods.	Data domains along with inner or internal boundaries can be better tested.
Example: search something on google by using keywords	**Example:** by input to check and verify loops

Black Box Testing

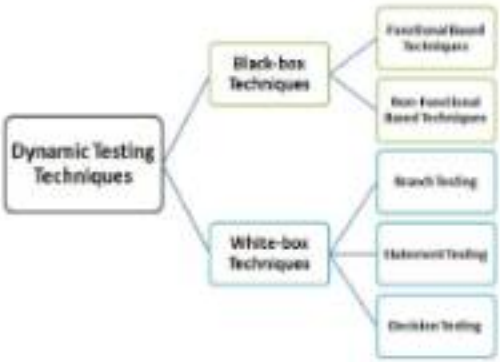
Types of Black Box Testing:

- A. Functional Testing
- B. Non-functional testing
- C. Regression Testing

White Box Testing

Types of White Box Testing:

- A. Path Testing
- B. Loop Testing
- C. Condition testing



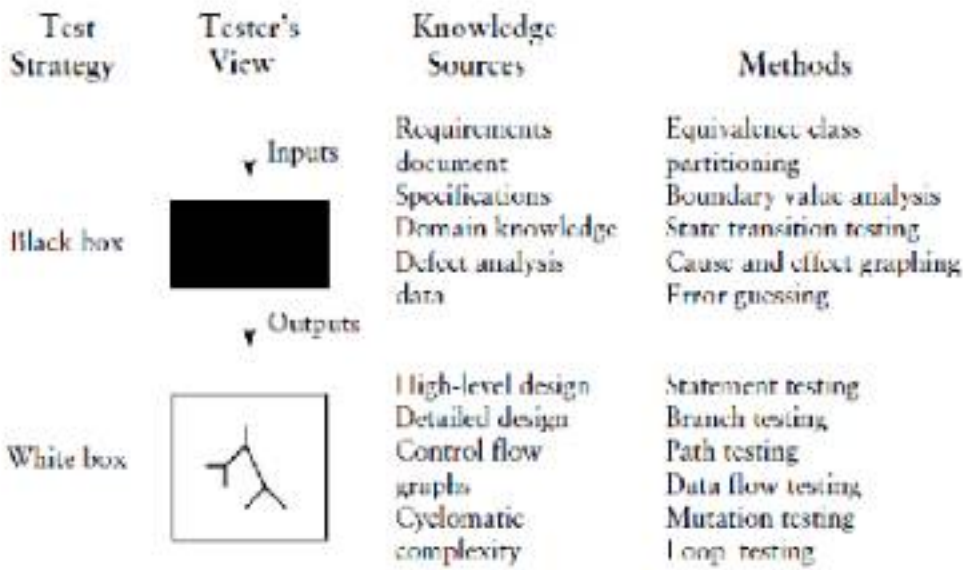
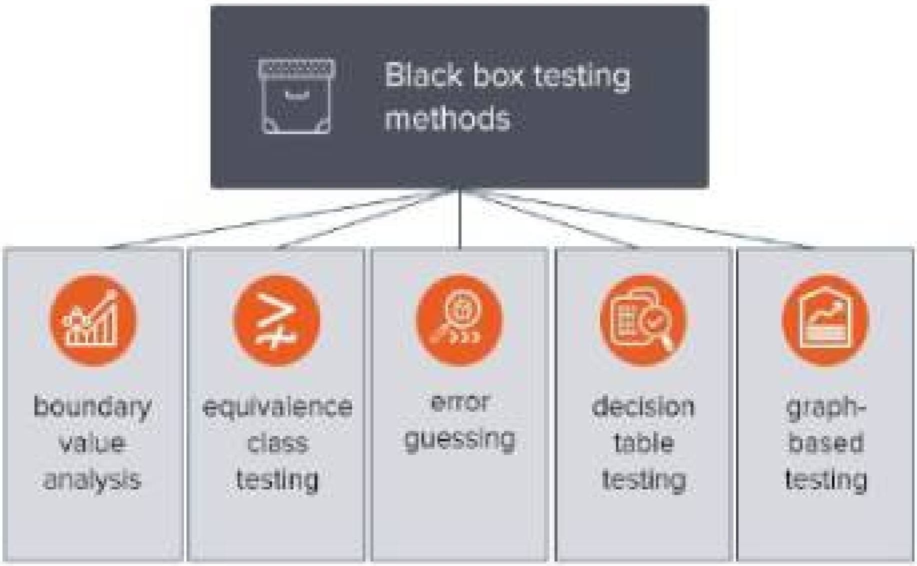


FIG. 4.1

The two basic testing strategies.

Black box testing == data driven testing = input/output testing

White box testing == logic drivern testing = Logic-Coverage Testing



BVA = Boundary value analysis based on verification of only extreme boundary values, e.g., maximum, minimum, and typical (Eg lower boundery 18 higher 56)

Age: [Enter Age] *Accepts value 18to 56

BOUNDARY VALUE ANALYSIS		
Invalid (min -1)	Valid (min, +min, -max, max)	Invalid (max +1)
17	18, 19, 55, 56	57

Name [Enter Name] *Accepts characters length (6 - 12)

BOUNDARY VALUE ANALYSIS		
Invalid (min -1)	Valid (min, +min, -max, max)	Invalid (max +1)
5 characters	6, 7, 11, 12 characters	13 characters

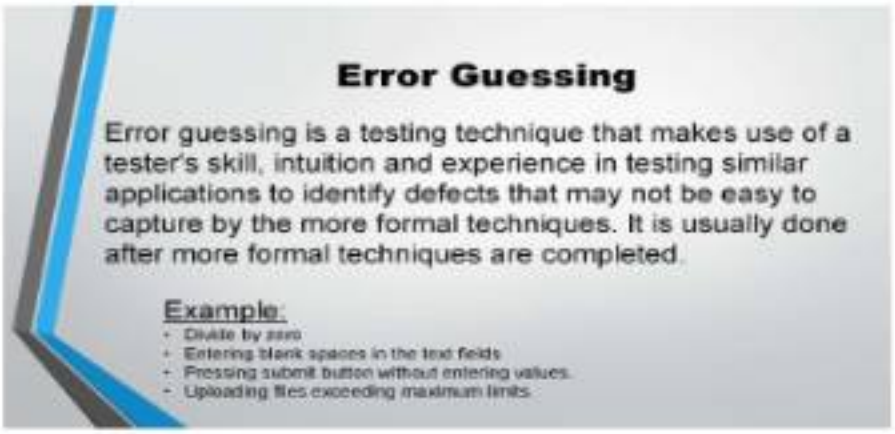
ECP = Equivalence class testing based on checking one value from each partition

Enter OTP [ ] *[illegible]

Equivalence Partitioning			
Invalid	Invalid	Valid	Valid
Digits–7	[illegible]	[illegible]	Digits–4
[illegible]	[illegible]	[illegible]	[illegible]

1 2 3 4 → 4 partition

ERROR Guessing = based on the previous experience of a QA engineer also called experience based testing



Decision table testing based on a tabular representation of combinations of inputs and

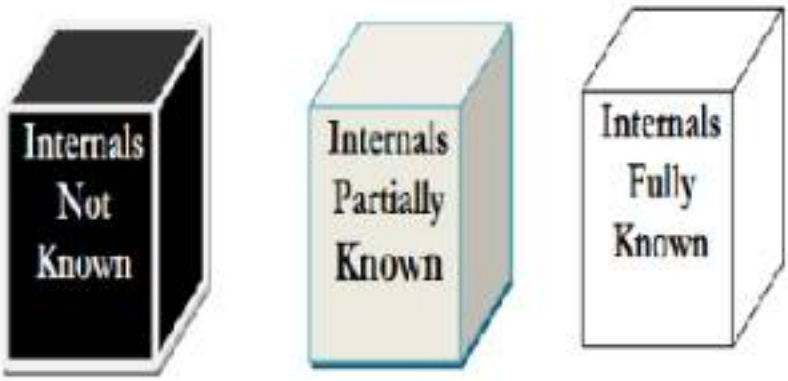
correspondent system behavior

depends of conition

CONDITIONS	1st Input	2nd Input	3rd Input
Salaried?	Y	N	Y
Monthly Income > 25000	N	Y	Y
IT Returns available?	Y	N	Y
ACTIONS			
Loan Eligibility	N	N	Y

Depends of condition or decision loan is given

Graph-based testing where a test case is written for each graph that represents the object



Comparison between the Three Testing Types

	Black Box Testing	Grey Box Testing	White Box Testing
1.	The Internal Workings of an application are not required to be known	Somewhat knowledge of the internal workings are known	Tester has full knowledge of the internal workings of the application
2.	Also known as closed box testing, data driven testing and functional testing	Another term for grey box testing is 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
3.	Performed by end users and also by testers and developers	Performed by end users and also by testers and developers	Normally done by testers and developers
4.	-Testing is based on external expectations -Internal behavior 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
5.	This is the least time consuming and exhaustive	Partly time consuming and exhaustive	The most exhaustive and time consuming type of testing
6.	Not suited to algorithm testing	Not suited to algorithm testing	Suited for algorithm testing
7.	This can only be done by trial and error method	Data domains and internal boundaries can be tested, if known	Data domains and internal boundaries can be better tested

White box testing

is an approach that allows testers to inspect and verify the inner workings of a software system—its code, infrastructure, and integrations with external systems

White box testing techniques

There are many ways you can analyze software with white box testing. Most testers will use a process called code coverage analysis to eliminate gaps in the testing of the code. There are a variety of techniques you can use to accomplish this, including:

Statement coverage:

This technique ensures that each line in the code is tested at least once to find faulty code more easily.

Branch coverage:

Using this technique, each possible path or decision point of a software application is checked for accuracy.

Condition coverage:

All individual conditions are checked.

Multiple condition coverage:

All imaginable combinations of all the conceivable condition outcomes are tested at least once.

Basis path testing:

Control graphs are created from either flowcharts or code. Cyclomatic complexity is then calculated to define the number of independent paths so that the minimum number of test cases can be designed for each path.

Flow chart notation:

This technique uses a directed graph made up of nodes and edges, where each node represents a decision point or sequence of statements.

Cyclomatic complexity:

This is the measure of a software's logical and cyclomatic complexity. It is used to define how many independent paths are present.

Loop testing:

Loops are commonly used in white box testing and are fundamental to many algorithms. Problems are often found at the beginning or the end of a loop. Loop testing can be divided into simple loops, nested loops and concatenated loops.