Manual testing

Contents:

SDLC (Software development life cycle)
- i. Waterfall Model
- ii. Spiral model
- iii. V model
- iv. Prototype model
- v . Agile model

Software testing
- i. **White box testing**
  - Path testing
  - Conditional testing
  - Loop testing
  - WBT from memory point of view
  - WBT from performance point of view
- **ii. Grey box testing**
- **iii. Black box testing**
  - Functionality testing
  - Integration testing
  - System testing
  - Acceptance testing
  - Smoke testing
  - Ad-hoc testing
  - Regression testing
  - Performance testing
  - Compatibility testing
  - Globalization testing
  - Exploratory testing

Test Plan

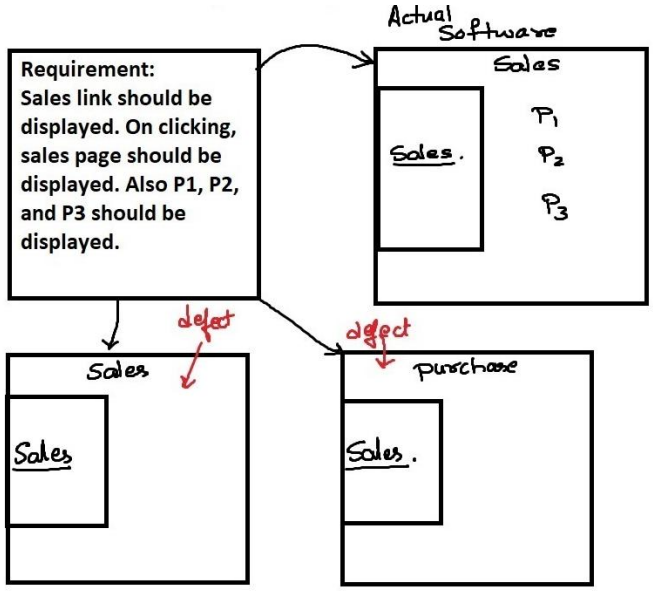
Defect tracking/ Defect life cycle

STLC (Software testing life cycle)

DEFECT

What is defect?

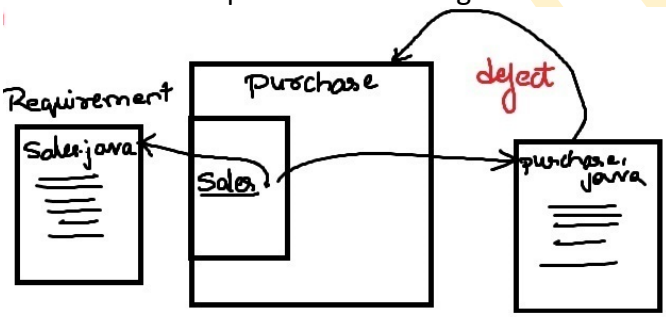
Any feature which is not working according to customer requirement is called as defect.
OR
Deviation from the requirement is called as defect.



Why do we get defects?

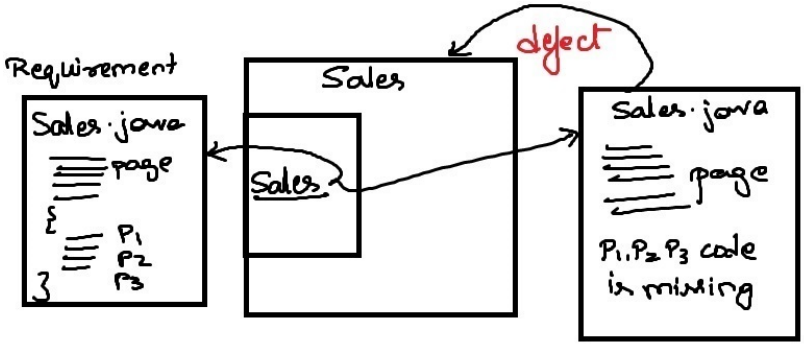
1) Wrong implementation

When the developer writes a wrong code for the feature.



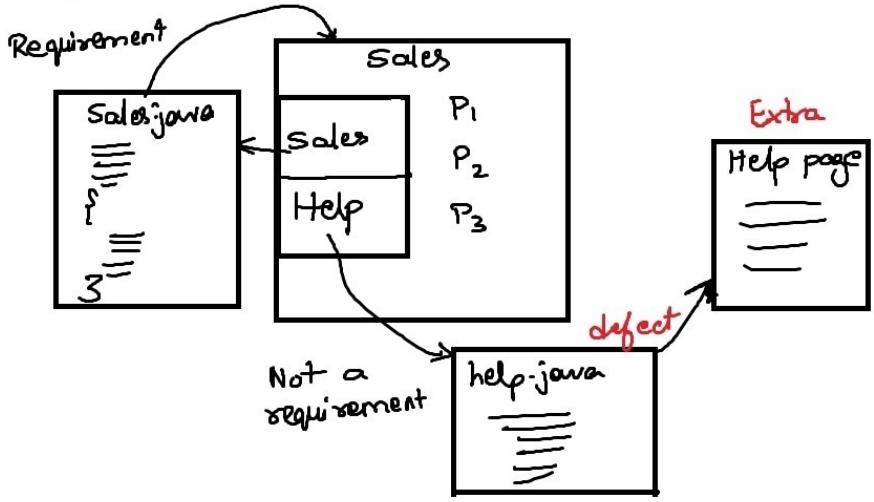
2) Missing implementation

When the developers misses to write the code for a feature.



3) Extra Implementation

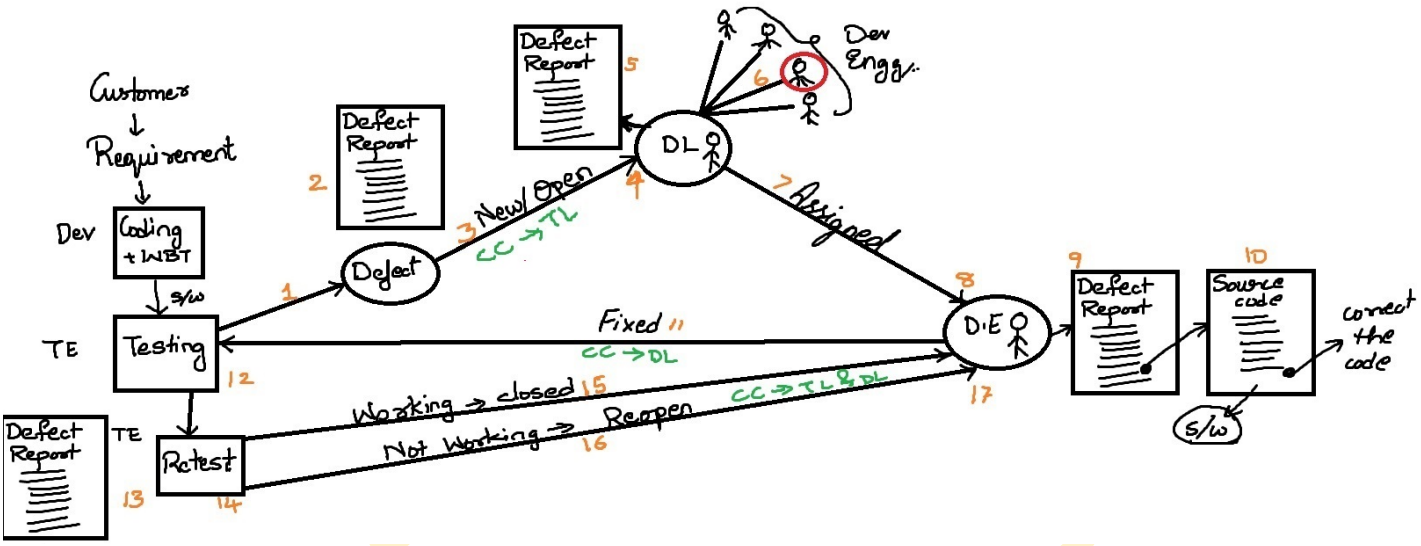
When the developers write the code for a feature which is not mentioned in the requirement.



What is the difference between defect, bug, error and failure?

DEFECT: Deviation from the requirement specification is called as defect
BUG: It is an informal name given to defect
ERROR: Mistake done by the developer in the code(program) because of which, we are unable to compile the program and run the program
Compile time error: We get this error because of mistakes in th syntax
Run Time error: We get this error because of the logical mistakes
FAILURE: The defects will lead to failure. One defect may lead to one or multiple failures

Defect Tracking Process



Roles of the TE, Developer and DL in Defect tracking process

Test Engineer
1) TE finds the defect
2) Prepares defect report
3) Puts status of defect as new/open
4) Sends the report to the development lead

Development Lead
1) DL reads the report and understands the problem
2) Identifies the developer who did the mistake
3) Changes the status to assigned
4) Sends it to development engineer

Development Engineer

1) DE reads the report and understands the problem
2) Goes to source code and fix the code
3) Change the status of defect to fixed
4) Sends the report to TE with cc to DL

Test Engineer

1) TE reads the report and understands the problem
2) Retests the fixed bug
3) If the defect is fixed, the status of defect will be changed to closed
4) If the defect is not fixed, the status of the defect will be changed to reopen
5) Send the report back to development engineer with cc to TL and DL

Severity

It is decided based on the impact of the defect on th customer business at different levels. That is Blockers, Critical, Major and Minor

Example-1:

Case-1:
Sender is unable to send the money to another user. This is a blocker defect
Case-2:
Money is transferred to receiver but it is not displaying data and time. This is critical
Case-3:
Money is transferred, date and time is also displayed, but the sender does not see the confirmation message. This is major
Case-4: The confirmation spelling in the confirmation message is not correct. This is minor

Example-2:

Case-1: Click on compose button and blank page is displayed. This is blocker
Case-2: Click on compose and attachment is not working. This is critical
Case-3: Click on compose and cancel button is not working. This is major
Case-4: Click on compose and send button colour is faded. This is minor

Priority

It is the importance given to the defect
-> How soon the bug must be fixed
-> Which one should be fixed first
There are different levels
High -> P1
Medium -> P2
Low -> P3

Generally all the blocker and critical defects are high priority. All major defects are medium priority and minor defects are low priority. But rarely critical can become low priority and minor can become high priority

Difference between Severity and Priority

Severity	Priority
It is decided based on the impact of defect on customer business	It is decided based on the importance given to the defect
The different levels are blocker, critical, major, minor	The different levels are high, medium, low
It is driven by functionality	It is driven by business value

Why we should not wait for the permission from the test lead to send the bug to the development lead?

1) There will be delay in communicating the defect report
2) As a test engineer, we will be knowing the module in depth, so better take self decision and send the defect report to DL with cc to TL

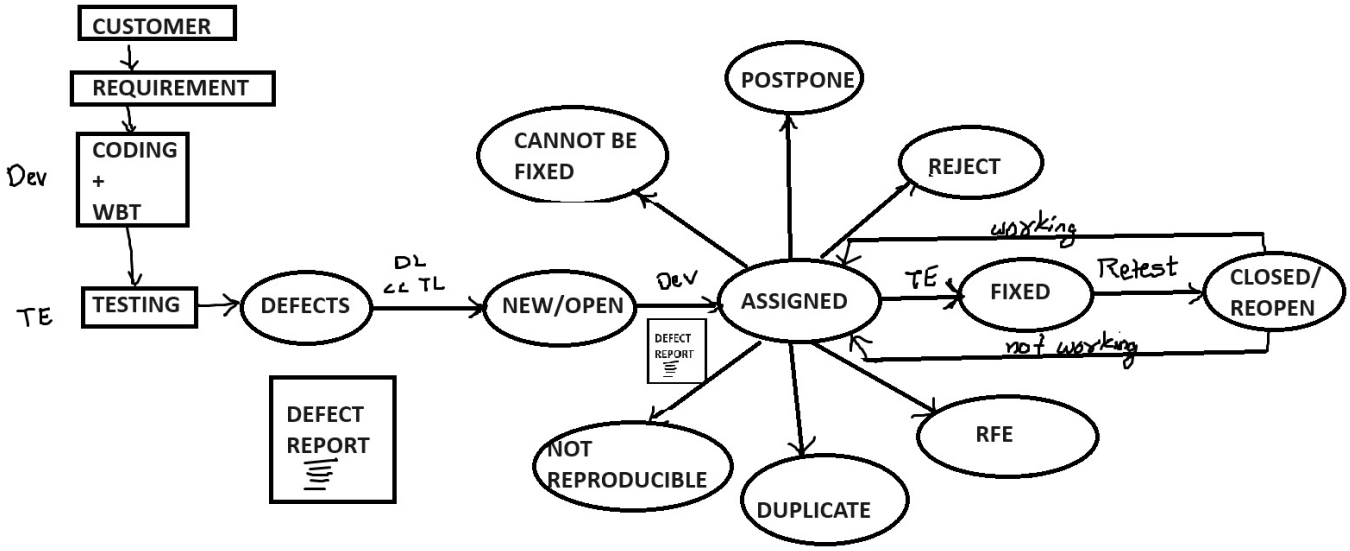
Why should we cc the bug report to the TL?

1) TL is the one who will be attending the customer management and development meetings. SO he should be aware of all the defects in the projects to avoid embarrassment
2) To get the visibility that we are working

As soon as we get the defect, we should send it to DL. Why?

1) Chances are there that we may forget
2) Developers will get sufficient time to fix that defect
3) Some other TE working in same module may send the defect first

Defect Life Cycle



When the TE raises the defect and the DL assigns it to the developer, the apart from fixing the defect, the developer may put the defect in to the following status

1) Reject
2) Cannot be fixed
3) Postpone
4) Not Reproducible
5) Duplicate
6) RFE- Request for Enhancement

REJECT

- When the TE has misunderstood the requirement
- When the TE has not installed the software properly
- When the TE is referring to old requirement
- When developers add an extra feature and TE reports it as defect

Cannot be fixed
- If the technology is not supporting
- If the TE finds the defect in the root of the software, when fixed the complete software will be affected
- If the cost of fixing the defect is more the defect itself

Postpone
- If the TE finds a minor or major defect at the time of release
- Due to insufficiency of time for fixing the defect

Not Reproducible
- Because of mismatch in platform (OS, browser, version, settings)
- Because of improper defect report
- Because of test data mismatch
- Because of build mismatch
- Because on inconsistent defect

Duplicate
- When two TE's testing the same module, raise the same defect

RFE- Request for Enhancement
- If the TE finds the defect in the software and if it is not a part of the requirement, then it will be set as RFE

How to avoid duplicate bugs?

Before the TE prepares a defect report and logs the bug, he should make sure that it is not a duplicate and should search in the defect tracking tool by entering certain keywords of the defect
Note: If the TE finds the defect after searching in DTT, then if the defect is in
1) New/open status -> Don't log
2) Assigned status -> Don't log
3) Fixed status -> Reopen
4) Closed status -> Create new defect

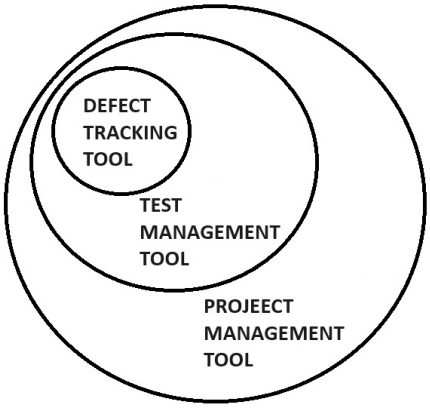
How will you convince it is a bug when the developer says that the bug is not reproducible?

1) Send the test data, test platform (OS, browser, version), also the credentials
2) Send the screenshot along with the proper steps (procedure) to reproduce the bug
3) Send the screen recording of the defect
4) Give a demo to the developer about the defect
5) Inform to test lead about the happenings

How to track defect manually?

1) Find the defect
2) Make sure that the defect is not duplicate
3) Prepare the defect report
4) Send the report to the development lead
5) Manage the defect life cycle

Management Tools



Project management tool
Manages and maintains the resources and source code
Test management tool
Manages and maintains the test cases and test cycle
Defect management tool
Manages and maintains the defects

Defect tracking tools

Defect tracking tool is a software which is mainly used to track the defects, store the defects and communicate the defects to the developer in an organized way.
1) Bugzilla
2) Bugzero
3) Buggini
4) JIRA
5) ALM
6) MANTIS

Defect masking

One defect masking or hiding another defect is called as defect masking

Defect seeding

Introducing defects to the software intentionally to test the efficiency of the TE is called as defect seeding

Defect Cascading

One defect triggering / causing another defect is called as defect cascading

Defect Leakage

Defect missed by the TE and found by the customer in the production is called as defect leakage

Defect Clustering

Defects which are accumulated in a particular module is called as defect clustering

Defect triage

When the time is very less, and there are too many pending defects to be fixed, then we categorize the defects in to different status
1) All the critical and blocker defects for the customer business will be moved to ASSIGN
2) Those defects, which has to be fixed but not immediately are moved to POSTPOME
3) Few defects which are not very important are moved to CANNOT BE FIXED
This type of grouping the defects is called as DEFECT TRIAGE
The meeting in which the defect triage is performed is called as DEFECT TRIAGE MEETING

TEST CASE

What is test case?

It is a document which covers all the possible scenarios for specific requirement

It has got different sections like

Step Number	Description	Input	Expected Result	Actual Result	Status	Comments

What is the difference between Test case and Test scenarios?

Test Cases	Test Scenarios
Detailed document of the scenarios which helps TE to test the application	High level documentation of all customer business workflow according to CRS
Test cases tell us how to test the software	Test scenarios tell us what to test in the software
One test case can have one test scenarios or many test scenarios	Test scenarios will not have test cases
Test cases are derived from the test scenarios	Test scenarios are derived from the requirement
Using the test cases we execute and find defects	Using test scenarios we develop test cases

Why we write the test cases?

1) We write the test cases to have better test coverage
    a) When the requirement comes in, developers are busy in building the product, same time TE will be free, so they should be identifying all the possible scenarios and document it
    b) When the build comes to, we can start or we can spend time in executing the scenarios immediately because the test cases are already written
2) To have consistency in test execution
    a) It means if we have documented the scenarios, we can make sure that we can execute all the scenarios in all the test cycles or sprints or releases . Will not miss testing any scenarios
3) It depends on process rather than person
4) To avoid training on every new engineer on the product or on the requirement
5) Test case is the only document which acts like proof for customers, development team and manager that we have covered all possible scenarios
6) Test case acts like a base document for writing automation scripts. If you refer the test case and write automation script, you can ensure that same amount of coverage in automation scripts also.
7) If the test cases are documented, there is no need for remembering the test scenarios
8) If you have documented the test case, test execution will happen in organized way
9) If you have written test case, time taken to execute it will be less

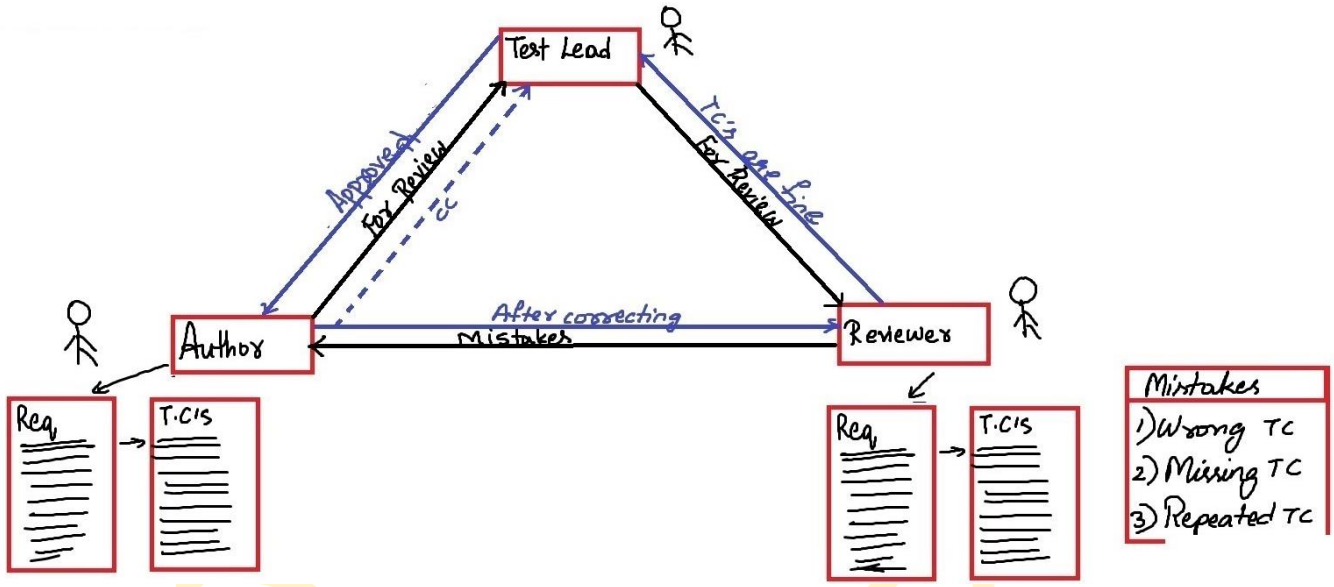
When do we write the test case ?

1) When the developers are busy in building the product, testing team should write the test case
2) When the customers add features, the test cases should also be updated
3) When the developers modify the features, there should be modification done in the test cases also

What is the drawback of seeing the requirement and testing the application ?

1) There will be no consistency in the test execution (The way we perform testing over a period of time will not be the same)
2) Quality of testing will depend on memory power of TE
3) Quality of testing depends on mood of TE
4) Quality of testing varies from person to person

Test case Review Process



On what basis we assign someone to review the test case?

1) If there is a person working on similar module
2) There is a test engineer who has worked on same module in previous project
3) There is a senior TE , in the project since beginning he knows every corner of the product, assign it to him
4) There is a TE highly responsible and smart assign it to him, he will find more mistakes and he will understand
5) Assign it to any one who understand the requirement and test case very fast and identify more mistakes

Review Ethics

1) Always review the content and not the author
2) While reviewing try to find only mistakes not the solution for it
3) Even after the review, if there is any mistake both author and reviewer are responsible

Why we review the test case? OR If I give you the test cases, what is the approach to review the test case? OR What kind of mistake will you find while reviewing the test cases?

1) We look in to the header of the test case and try to understand the requirement for which the test case is written then we try to find
    a) Missing scenarios
    b) Repeated scenarios
    c) Wrong scenarios
2) We will check whether scenarios are organized properly or not, so that when it is executed it will take less time
3) We will check whether it is simple to understand or not so that when it is given to new engineer, he should be able to execute it without any questions
4) We look In to the header of the test case and try to check whether
    a) All the attributes are covered or not
    b) We try to check whether all attributes are having relevant content or not
5) We check whether test case format is according to the standard defined in the project

Test case Review Template

Sl. No	Test case name	Step number	Reviewer comments	Severity	Author comments

Test case Design Technique

There are three techniques

1) Error Guessing Techniques
2) Equivalence Partitioning
3) Boundary value analysis

1) Error Guessing Techniques

Guessing all the possible errors or defects and deriving the scenarios

We guess the errors based on

a) Requirement
b) Experience
c) Intuition

2) Equivalence Partitioning

It has 2 methods

a) Pressman method

i. If the input is range of values, then design the test cases with 1 valid and 2 invalid values.
Ex: Amount Transfer -> Range is 100 to 10000. Inputs are 500 (valid), 90 and 10001(invalid)

ii. If the input is set of values then design the test cases for 1 valid and 2 invalid values
Ex: Product ID: (Printer -10, Scanner-20, Webcam-30, Mouse-40). Inputs are 20 (valid), 5 and 60(invalid)

iii. If the input is a boolean, assign the test case for both true and false values
Ex: When we test for checkbox or radio button, we should test the application for both check (true) and uncheck (false) conditions

b) Practice method

If the input is range of values then divide the range in to equivalent parts and test for all the values and make sure that you are testing for at least 2 invalid values
Ex: Amount Transfer -> Range is 100 to 10000 with the deviation of interest for 100 to 5000 as 1% and 5001 to 10000 as 2%
The equivalent parts are 100, 5000 and 10000. The 2 invalid inputs are 50 and 10001

3) Boundary value Analysis

If the input is range of values between A to B, then design the test cases for A, A+1, A-1, B, B+1, B-1 values
Ex: Amount transfer -> Range is 100 to 10000, then design the test cases for 100, 101, 99, 10000, 10001 and 9999, which covers 4 valid and 2 invalid values

Test case optimization

1) If the input is range of values -> Use BVA and EG
2) If the input is range of values with deviation -> Use practice method and EG
3) If the input is set of values -> Use Pressman method and EG
4) If the input is boolean -> Use Pressman method and EG

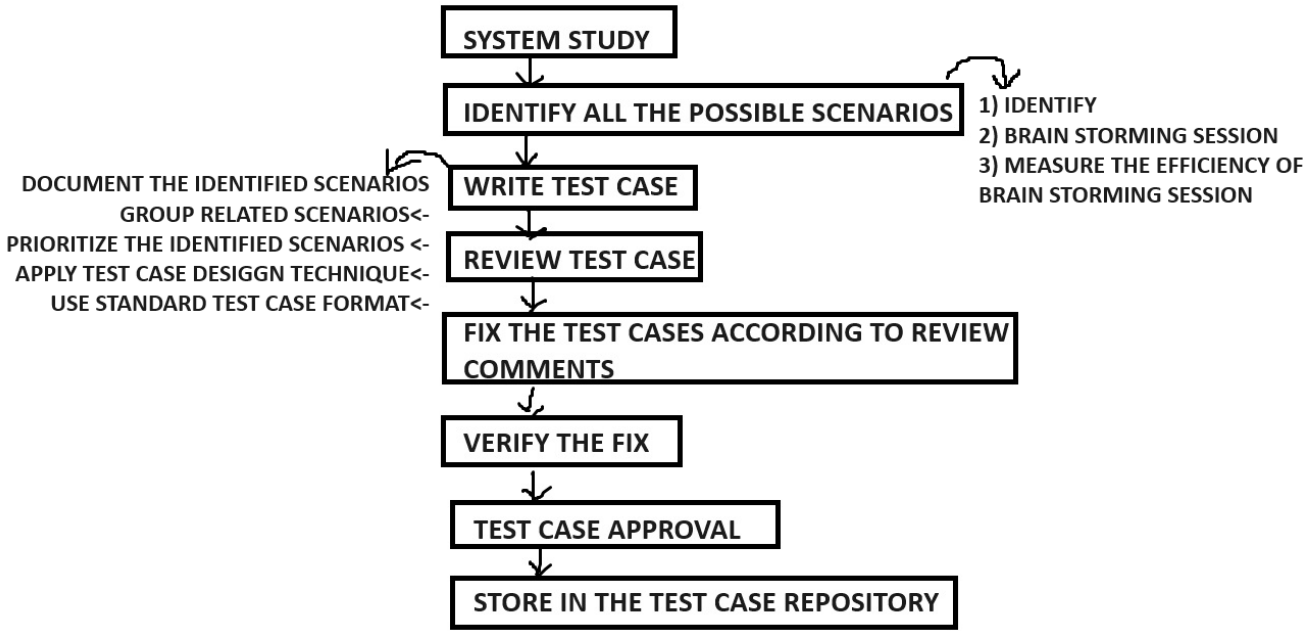
Test case template

Rules for writing the test case

TEST CASE TEMPLATE					
1) HEADER					
2) BODY					
3) FOOTER					
HEADER			1		
TESTCASE NAME/TESTCASE ID	UNIQUE NAME GIVEN TO EVERY TEST CASE. SYNTAX: PROJECTNAME_MODULENAME_TESTCASEID	AM-LG-01	1.1		
REQUIREMENT NUMBER	TAKEN FROM THE SRS	2.3	1.2		
RELEASE NAME	NAMES OR RELEASE NUMBERS WHICH COMES IN VERSIONS	RELEASE-3 V-5	1.3		
MODULE NAME	FROM REQUIREMENT	LOGIN			
PRE-CONDITION	1)IT IS THE SET OF ACTIONS OR SETTINGS WHICH SHOULD BE TAKEN CARE BEFORE THE TEST CASE EXECUTION. THERE ARE TWO TYPES OF CONDITIONS. IMPLICIT AND EXPLICIT 2)IMPLICIT NEED NOT BE MENTIONED IN THE TEST CASE AS IT IS UNDERSTOOD EX: PRESENCE OF LAPTOP, PRESENCE OF POWER, PRESENCE OF INTERNET, INSTALLATION OF SOFTWARE 3)EXPLICIT CONDITIONS SHOULD BE MENTIONED IN THE TEST CASE EX: SETTINGS(BROWSER OR DEVICE), PERMISSIONS, URL, UN AND PASSWORD	URL UN PWD			
TEST DATA	THE DATA CREATED BY THE TE IN THE APPLICATION ITSELF BEFORE EXECUTING THE TEST CASE. USUALLY THE DATA IS KEPT READY BEFORE RECEIVING THE BUILD	URL -WWW.AMAZON.COM UN - ABC PWD -123			
SEVERITY	IT IS BASED ON THE COMPLEXITY AND IMPORTANCE OF THE MODULE TESTED	HIGH			
TESTCASE TYPE	FT,IT,ST,ADHOC,SMOKE				
BRIEF DESCRIPTION	SUMMARY OR THE HIGH LEVEL BEHAVIOUR OF THE FEATURE	THIS TEST CASE EXPLAINS HOW THE USER SHOULD BE LOGGING IN TO THE AMAZON APPLICATION			
TEST CASE EXECUTION HOURS	ESTIMATED TIME TAKEN TO WRITE AND EXECUTE THE TEST CASES	5 HOURS			
BODY					
STEP NUMBER	INPUT	EXPECTED RESULT	ACTUAL RESULT	STATUS	COMMENTS
SPECIFIES THE NUMBER OF TEST CASES	DATA /VALUES THAT IS TO BE ENTERED WHILE EXECUTING THE TEST CASES	THE RESULT WHICH SHOULD BE PRESENT AFTER EXECUTING THE TEST CASE.	THE RESULT WHICH APPEARS AFTER EXECUTING THE TEST CASE	THE STATUS IS THE RESULT OF EXECUTIO	REASON FOR FAILURE, DEFECT ID OR ANY OTHER INFO
FOOTER	1) OPEN THE BROWSER 2) ENTER THE URL - WWW.AMAZON.COM 3) ENTER THE UN AND PASSWORD -ABC AND 123 4) CLICK ON LOGIN BUTTON	HOME PAGE SHOULD BE DISPLYED	ERROR PAGE IS DISPLAYED	FAIL	AM-LG-D-007
AUTHOR NAME	THE ONE WHO WRITES THE TEST CASE	MYTHREYE			
REVIEWED BY	THE ONE WHO REVIEWS THE TEST CASE	GANESHA			
APPROVED BY	THE ONE WHO APPROVES THE TEST CASE	GANESHA			
APPROVED DATE	THE DATE WHEN THE TEST CASE WAS APPROVED	11-09-2023			

1) Before you start writing the test case come up with an option and select the best option. The options are with respect to the template or with respect to the test approach
2) Always use should be /must be/should not be/must not be in the expected result. Do not use maybe/may not be/ can be/cannot be..
3) Elaborate only those steps In which we have the focus. Do not elaborate all the steps
4) Always start writing the test cases with the navigation steps
5) Always write generic test cases and do not hard code the test cases
6) While writing the test cases, we should imagine the application
7) If you organize the scenarios properly, you can reduce the total number of steps
8) If something is covered in functionality test case, don't repeat the same thing in the integration test case. If some thing is covered in integration test case, don't repeat the same thing in system test case

Procedure to write the test cases



What is brain storming session?

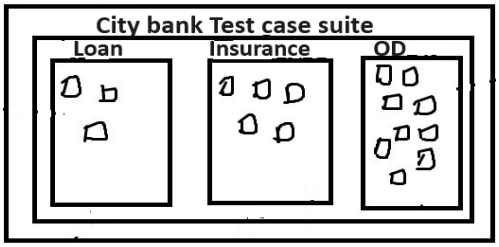
1. TE will present a demo on the feature(improved product feature) to the team.
2. He also presents the scenarios written for the features
3. The other testing fellow colleagues (TL also) identify the missing, wrong or repeated test scenarios

What is repository?

It is a centralized place where all the test cases are stored in the form of test suite

What is test suite?

Collection of group of test cases

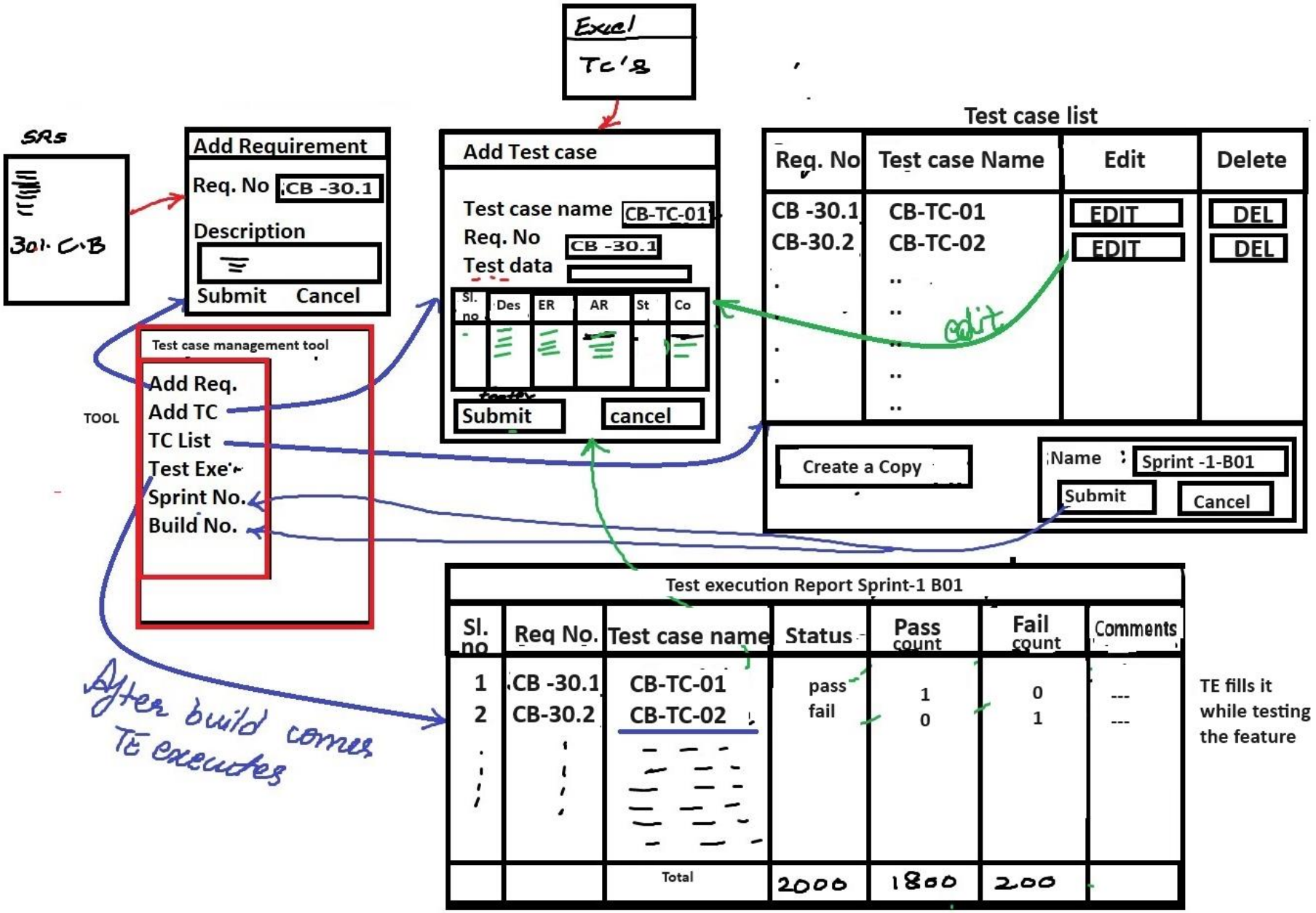


Test case management tool

It is a tool which is used to store the test cases in an organized manner in the centralized place.
Where do we write the test cases?

1. Excel
2. Word
3. Test case tools (JIRA, QC, ALM….)

Test case tool


Excel
Tc's
SRs
301. C-B
Add Requirement
Req. No CB -30.1
Description
Submit Cancel
Add Test case
Test case name CB-TC-01
Req. No CB -30.1
Test data
Sl. no
Des
ER
AR
St
Co
Submit
cancel
Test case list
Req. No
Test case Name
Edit
Delete
CB -30.1
CB-30.2
CB-TC-01
CB-TC-02
EDIT
EDIT
DEL
DEL
edit
Create a Copy
Name : Sprint -1-B01
Submit
Cancel
Test case management tool
TOOL
Add Req.
Add TC
TC List
Test Exe
Sprint No.
Build No.
Test execution Report Sprint-1 B01
Sl. no
Req No.
Test case name
Status
Pass count
Fail count
Comments
1
2
CB -30.1
CB-30.2
CB-TC-01
CB-TC-02
pass
fail
1
0
0
1

Total
2000
1800
200
After build comes TE executes
TE fills it while testing the feature

EXECUTION REPORTS GENERATED BY TOOL

Sl.no	Module name	Total TC's	TC's Executed	TC's not Executed	Total Pass	Total Fail	Pass %	Fail %
1	CA	400	400	0	340	60	85	15
2	AA	300	0	300				
3	LOGIN	200	0	200				
4	LOGOUT	100	0	100				
	TOTAL	1000	400	600	340	60	85	15

SUMMARY	CA	AA	LOGIN	LOGOUT

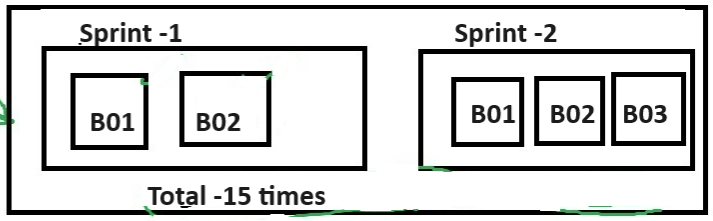
Create Account

Module(Story) wise execution report

Sl.No	Test case	Status	Comments
1	CB-TC-01	pass	...
2	CB-TC-02	fail	...
3			
4			
.			
.			
	Total	340	

Test case wise execution result

Total execution of TC's in all sprints



TRACEABILITY MATRIX

It is a document which makes sure that each and every requirement given by the customer has got atleast one test case written.

Types of Traceability matrix

1. Forward traceability matrix
2. Reverse Traceability matrix
3. Bi-directional traceability matrix

Forward TM

We will map the requirements with the test cases to make sure that each and every requirement has got at least one test case written

This is prepared before test case execution

Reverse TM

We will map the test cases written with the requirement. Once after the test case execution is done, to make sure that the product which is developed and tested is really according to customer requirement, to make sure that we are building Right product. This is done after the test case execution

Bi-directional TM

It is the combination of FTM and RTM

TEST PLAN

It is a document which drives all the future testing activities
It has 15 attributes
1) Objective
2) Scope
3) Test methodology
4) Test approach
5) Assumption
6) Risk
7) Back up plan/Mitigation plan/Contingency plan
8) Roles and Responsibilities
9) Scheduling
10) Defect tracking
11) Test Bed/ Test Environment
12) Entry and Exit Criteria
13) Test Automation
14) Deliverables
15) Templates

Objective

This section covers why we are preparing the test plan or what is the aim of preparing the test plan

Scope

This section covers features to be tested and not to be tested in future
EX: Gmail:
✓ Features to be tested (in scope)
Sign up, sent items, login, compose, all mails
Inbox-> In inbox delete mail is out of scope-> to be taken in next sprint)
✓ Features not to be tested (out of scope)
Help, activities

Anything in the requirement should be tested
Whenever you are preparing test plan in 2 or 3 sprints, certain old features must be included in to scope
Certain old features we must move to out of scope because it may not have impact because of new features
Note:
Test Strategy
The company management along with the higher authorities like PM, BA… create a document of planning a project. This contains many test plans.

Test Methodology

States what kind of testing should be done based on the type of application.
EX: Web security testing, globalization testing need not be performed for all applications

Test Approach

It explains how we have to go about and start testing the software
✓ Few companies write test cases alone
✓ Few just write high level scenarios
✓ Few go with scenarios and test cases

✓ Few also follow up with the flow chart

Assumptions

✓ Resource (human /technical) point of view
✓ Support point of view
✓ KT point of view

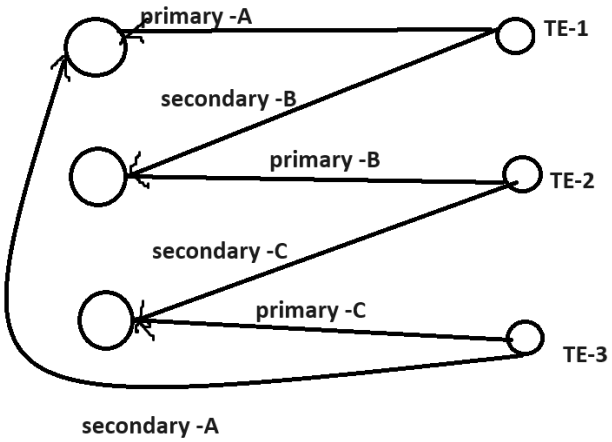
Assumptions that all these would be there till the product is delivered to production

Risk

When assumptions fail and when engineer misses important bugs

Back up plan

In order to avoid risk, we should have a backup plan



We are planning means we are assuming that we are gonna do many task based on that we promised customer because of few reasons if we are not able to do certain task that becomes a risk.

How to face the risk?

To face the risk, we have the plan, that plan is nothing but mitigation plan/backup plan
Everybody will be there in the project till the end is an assumption
Someone quits and new fellow might miss lot of defects and this is a risk
The backup plan is that new engineers should refer the test case and test product with secondary owner's help.

Roles and Responsibilities

This section covers what each engineer should do in different stages of test life cycle

Roles and responsibilities of TM

1. Write and reviews the test plan
2. Interact with testing team, dev team, PM and if needed customer
3. Handles issues and escalations
4. Approves release notes

Roles and responsibilities of TL

1. Write and reviews the test plan
2. Allocate work to TE and make sure that they are going to complete the task within the schedule
3. Consolidate the reports which are sent by every TE and communicate with the testing team, dev team PM and customer
4. He conducts impact analysis meeting

Roles and responsibilities of TE

1. He write the test cases
2. He reviews the test cases of other TE
3. He executes the test cases for his allocated features

Roles and responsibilities of ATE

1. Converts TC's in to automation test scripts
2. Executes the scripts
3. Updates the scripts

Scheduling

This section covers when exactly which activity should start and end according to company promised date and customer requested date

Defect tracking

This section covers, in future , when we find defects, how we should track it and also it covers what should be the procedure, status, severity and priority

1. Procedure to track the defects
2. Severity
3. Priority
4. Defect tracking tool used

Test bed/Test Environment

This section covers how we go about setting up the test environment in future

12.1 Procedure to install the build

12.2 Hardware

12.2.1 Server side

HP star cat 1500

12.2.2 Client side

6 computers with following configurations

-> 8 GB RAM

-> Quad RAM

-> Intel

12.3 Software

12.3.1 Server side

Os: Linux

Web server - TOMCAT

App Server - Web sphere

DB server - Oracle

12.2.2 Client side

OS: Win 10, Win 8

Browser - Mozilla and chrome

Entry and Exit criteria

Entry Criteria -> The list of criteria that should be met to start the activity

Exit Criteria -> The list of criteria that should be met to say that the activity is completed

Entry criteria for FT
1. WBT should be done
2. Test cases should be ready
3. Build should be installed with proper testing environment
4. Test data should be available
5. Resources should be available

Exit criteria for FT
1. Percentage of test cases executed should be 90%
2. Percentage of test cases pass should be 85%
3. It should not cross more than 10 critical bugs
4. It should not cross more than 40 major bugs
5. It should not cross more than 80 minor bugs

Entry criteria for IT
1. It should meet the exit criteria of FT
2. The IT test cases should be ready
3. Build should be installed with proper testing environment
4. Test data should be available
5. Resources should be available
6. Smoke and functionality should be completed

Exit criteria for IT
1. Percentage of test cases executed should be 90%
2. Percentage of test cases pass should be 85%
3. It should not cross more than 10 critical bugs
4. It should not cross more than 40 major bugs
5. It should not cross more than 80 minor bugs

Entry criteria for ST
1. It should meet the exit criteria of IT
2. The ST test cases should be ready
3. Build should be installed with proper testing environment
4. Test data should be available
5. Resources should be available
6. Smoke, functionality and integration should be completed

Exit criteria for ST
1. Percentage of test cases executed should be 90%
2. Percentage of test cases pass should be 85%
3. It should not have any critical bugs
4. It should not cross more than 10 major bugs
5. It should not cross more than 40 minor bugs

The criteria for entering the exiting the testing might vary according to company standards

Test stop criteria
1. All the features requested by the customer should be ready
2. If all the end to end business scenarios are working fine
3. If there are zero blockers and zero critical bugs and there are some bugs which is there within the acceptable limit set by the customer
4. The product should be tested in the environment which is similar to the test environment

When to not stop testing?

1. Too many blockers and critical defects
2. Many of the end to end critical business scenarios not working
3. If it is crossing the budget and also schedule

Test Automation

This section tells what features to be automated and what not and the complete automation technology

1. Features to be tested
2. Features not to be tested
3. Automation frame work - DJANGO
4. Automation tool - Selenium

Deliverables

Covers which, output / documents that has to be provided by testing team at the end of test cycle

1. Test case
2. Traceability matrix
3. Test execution report
4. Defect report
5. Release notes
6. Graphs and Matrices

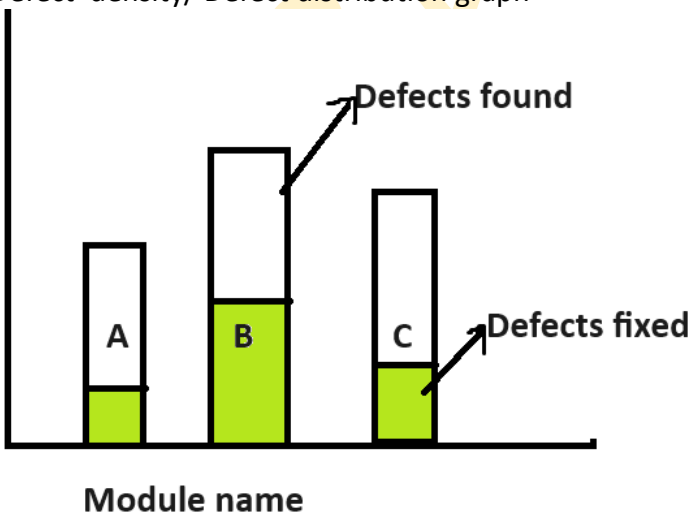
Release notes:

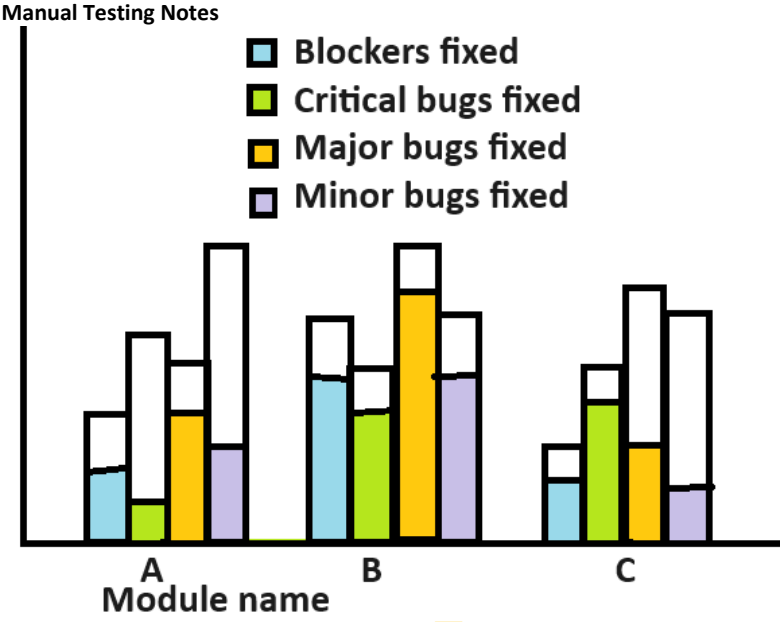
Along with the product, we send release notes to the customer that is called Release notes. It contains

1. List of known defects/ open defects which are there in the product
2. List of platforms in which product is tested
3. List of platforms in which the product is not tested
4. Number of bugs found in previous release and fixed in current release
5. List of features which are added /modified/removed
6. Version number and installation steps

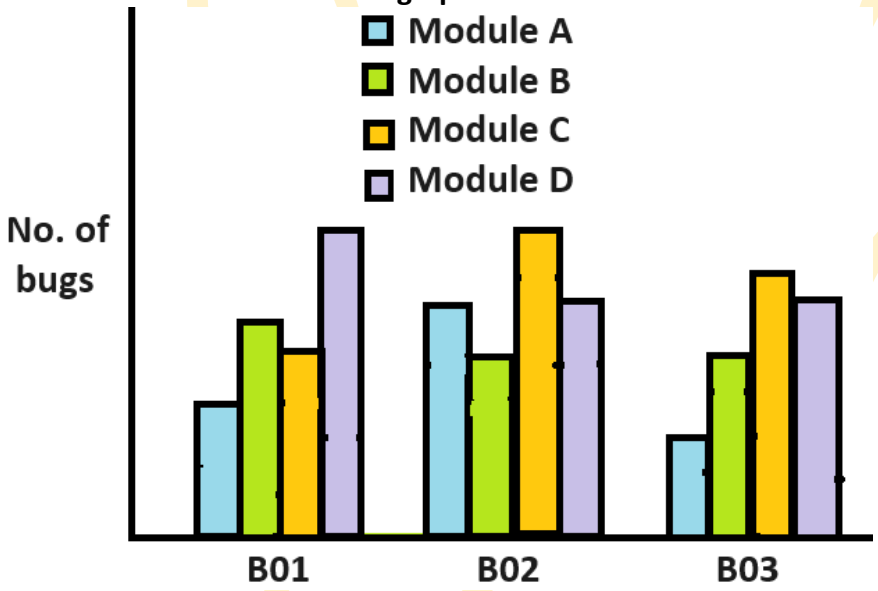
Graphs and Metrics

Defect density/ Defect distribution graph

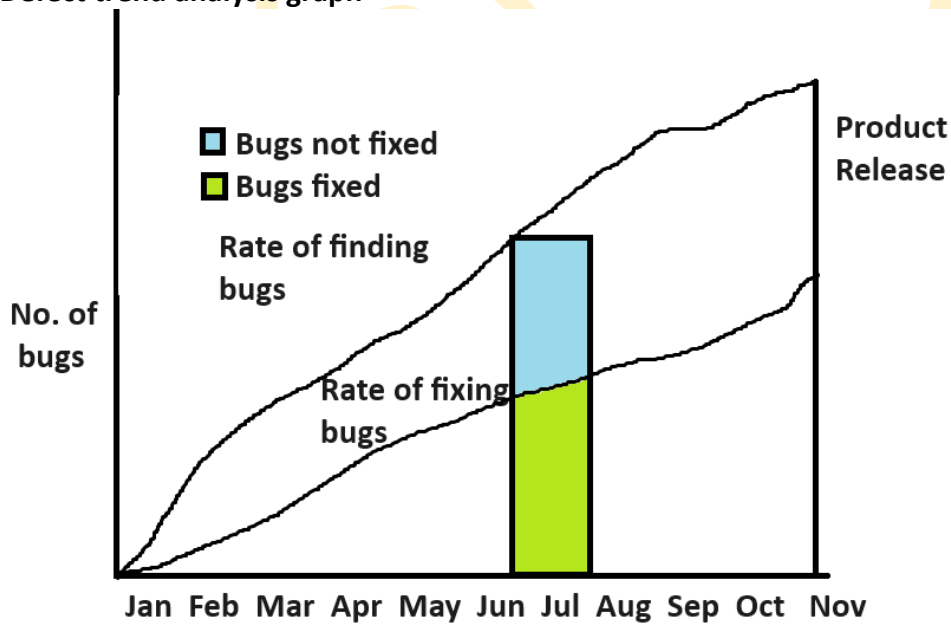




Build wise defect distribution graph



Defect trend analysis graph



Module wise Defect distribution metrics

		Critical		Major		Minor	
Sl. No	Module name	Found	Fixed	Found	Fixed	Found	Fixed
1	A	40	37	80	69	110	89
2	B	80	70	150	139	200	167
3	C	60	55	100	96	140	110
4	D	70	64	90	87	120	101

TE wise Defect distribution metrics

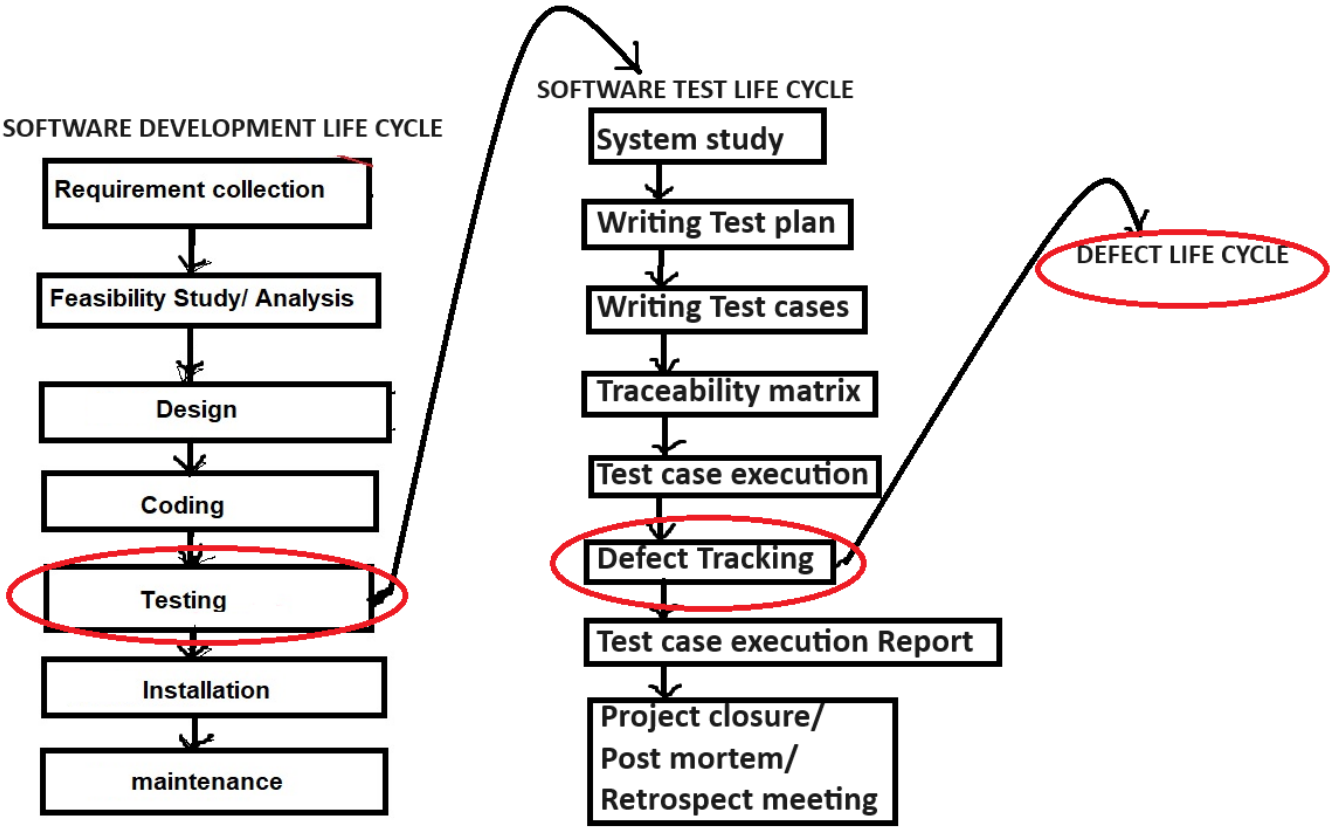
		Critical		Major		Minor	
Sl. No	TE name	Found	Fixed	Found	Fixed	Found	Fixed
1	Dinga	40	37	80	69	110	89
2	Dingi	80	70	150	139	200	167
3	Penga	60	55	100	96	140	110
4	Pengi	70	64	90	87	120	101

It is the past data collected between the two. So that we can analyse and take better decisions to perform future activities

Templates

This section covers formats for all the documents that we are going to prepare in the complete cycle

STLC- Software test life cycle



Important Interview Questions

1) What are the attributes of the test case/defect report?
2)what is test data?
3) what is a test script?
4) what is defect priority and severity?
5) What are critical bugs?
6) what is specification-based testing(BBT)?
7) What is the advantage of testing design in the early life cycle?
8) The difference between retesting and regression testing?
9) what type of testing will you do when you get a patch?
10) How will you test the login page of a web application?
11) How will you test the web application?
12) what is the need for compatibility testing?
13) how to track defects/bugs manually?
14) why does an application have bugs?
15)for what type of testing will you write test cases?
16)what do you like about Manual testing?
17) what do you dislike about Manual testing?
18) The difference between formal testing and informal testing?
19) what do you mean by WBT AND BBT?
20) types of test coverage techniques?
21)bug triage meeting
22) impact analysis meeting
23) what is a test bed?
24) what is the procedure of manual testing? (STLC)25) what is the role of documentation in Manual testing?
26) bug life cycle?
27) difference b/w test server and production server?
28) what is a test closure document?
29) difference between positive testing and negative testing?
30) what is a pesticide paradox?
31) static and dynamic testing?
32) Why do we go for smoke testing?
33) what is the need to do smoke testing?
34) how do you determine when to stop testing?
35) how will you choose the test cases that are to be automated?
36) when you get a new build what activities you will do?
37) when is RTM prepared?
39) what is the procedure of the SDLC?
40)what happens when a developer is involved in testing?
41) what is the key phase in SDLC?
42) What is mute phase in SDLC?
43) in Spiral- model if you get requirement for next module and a major change in old module what will you do here?
50) difference between prototype testing and actual testing?
51) what happens when a test engineer is involved in fixing a defect?
52) difference between test scenarios and test cases?
53) when there is only one module is ready what type of testing will you do?
54) why do we get new defects in the old module?
55) what is the test cycle?
56) what is a patch?
57) what is a hot fix?
58) who will do acceptance testing?
59) what is alpha testing and beta testing?
60) what is the difference between a defect error bug and failure?

61) why we should not wait for the permission of TL to send a defect report to DL? 62) Why should we keep CC to TL?
63) who will give severity and priority?
64) Why will we get REJECT status ?
65) what is the issue not reproducible?
66) when the developer will give "defect can't be fixed " status?
67) who will give "RFE”?
68) What is a defect cascading?
69) what is defect clustering?
70) what is defect leakage?
71) can we go for automation in smoke testing?
72) difference between smoke and sanity?
73) what is the need to do ADHOC testing?
74) Why we will not do ADHOC testing in the early stage only?
75) what happens if we don't do comparability testing?
76) what happens if we don't write test cases?
77) after which activity TE will start writing Test cases
78) what is the key element of test cases?
79) what is the key element of the test plan?
78) How will you review the test cases?
79) what is the procedure to write test cases?
80) if there is no requirement how will you prepare traceability matrix?
81) difference between Traceability matrix and test case review?
82) what is product backlog meeting?
83) what do you mean by sprint plan meeting?
84) what is retrospective meeting?
85) what is agile methology?