SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

Requirements collection (document)

⬇

Feasibility study

- HR team (resources)
- Finance team (profitable)
- Project manager / Architect (technically it can be done or not)

⬇

Design

- High level design / module level (architect)
- Low level design / component (project manager)

⬇

CODING
(assign to developers)
Application will be ready.

⬇

TESTING
(assign to test engineers to find defects)
Send back to developers to make code changes , again tested till the app is stable .

⬇

INSTALLATION
(installation team will move application from company environment to client environment)

⬇

MAINTENANCE
To solve issues

SDLC. stands for software development life cycle
It is a step by step procedures to develop an app
It consists of various phases such as requirement collection,feasibility study , design, coding, testing, installation, maintenance

Requirement collection
In this phases the business analyst (BA) goes to customer place and collects the business need of the customer in the form of documents

Feasibility study
In this phase a team of high level people sit together and decide whether the project of technically doable or not on the basis of some factors
Hr team : will decide the resources are available or not
Finance team : decides profitable or not
Where PM and arch level : will decide technically doable or not

Design
Design means blue print off the app
Design are of two types

A) high level design
   It is also known as module level design and it is done by the architect
B) low level design
   It is also known as component level design and it done by the project manager

Coding
In this phase the developer start writing the code by choosing one programming language this process of writing the code by the developers continue until the app is ready

Testing
Once the app is ready the test engineers start checking the app . While checking the app the TE may find some defects .
All these defects are handed over to the developers
The developer does the necessary code changes to fix the defects and give it back to the test engineers .
This process of checking the app i.e. finding the defects getting fixed by the developers continues until the app is stable.

Installation
In this phase the final stable product is moved from the company environment to the client environment and it is done a separate team known as installation team

Maintenance
Once the app is deployed into the client environment , the customer starts using the app , while using the app the customer may encounter some issues in order to overcome such issues and get it resolved immediately One TE and developer deployed to the client environment for a period of time .

MODELS OF SDLC

- waterfall model
- Prototype model
- Spiral model
- V model
- Hybrid model
- Agile methodology

WATERFALL MODEL

Requirement collection
⬇
Feasibility study
⬇
Design
⬇
Coding
⬇
Testing
⬇
Installation
⬇
Maintenance

The Waterfall model is the base model of SDLC and all other models are derived from it.
In this model once we connect the requirements we freeze it and thereafter execution happens in a sequential order i.e, the output of one phase is the input for the next phase.
The changes given by the customer in between are implemented in the next release .

Drawbacks of waterfall model

- In between requirements changes are not allowed
- No parallel deliverables or no two teams working at a time
- Review process does not exist
- Downward follow of bugs
- Developer used to do the testing

Advantages

- It is the best suitable model for life critical and machine critical application .

PROTOTYPE MODEL

Requirements collection
⬇
Feasibility study
⬇
Prototype development - UI/ UX engineers
Prototype testing - BA
Review and approval - Client
⬇
Design

⬇
⬇
Coding
⬇
Testing
⬇
Installation
⬇
Maintenance

We go for prototype models whenever the customer is new to the software .
In this model we develop a replica of the application and get it approved by the client before moving it to actual development of the application.
In between changes are allowed.

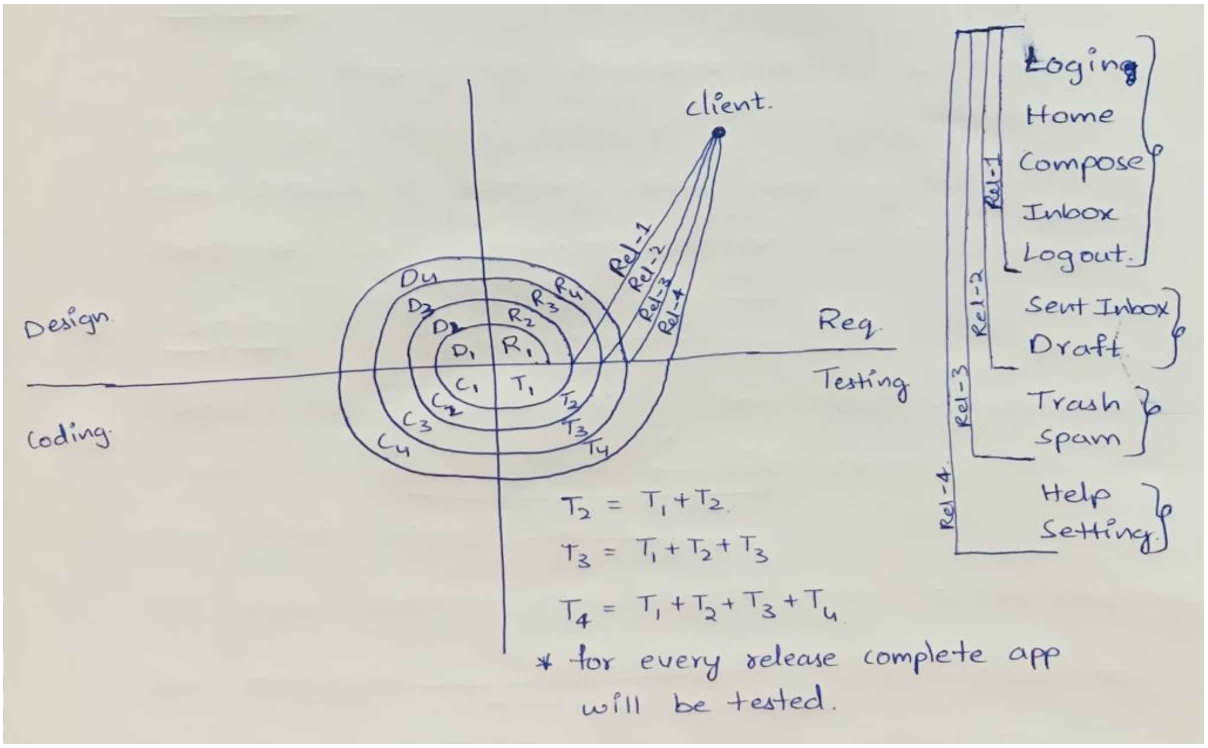
Advantages

- customer satisfaction existed or no customer rejection
- Prototype once developed can be reused for multiple similar customer

Disadvantages

- review process does not exist
- Time consuming model
- No parallel deliverables

SPIRAL MODEL



We go for spiral model whenever the modules are dependent on this model we pick few of the dependent modules develop it and hand it over to the customer so that customer start using it for their business parallely we develop features and hand it over to the customer in various intervals of time

Advantage

- Customer start using the application at very early stage
- In between changes given by customer are allowed

Disadvantage

- no parallel deliverables
- Review process doesn't exist
- In Between changes given but the customer are of two types

A) Minor changes

Minor changes are those changes due to implementation of which in the current release will not affect the existing feature such changes can be implemented in the same release

For eg : attachment is a feature already existing, but it may not accept zip files, in order to make zip files also to be accepted it is a minor change and can be done in the same release.

B) Major changes

Major changes are those changes due to implementation of which in the current release may affect the existing features such changes cannot be implemented in the same release they are implemented in the next release

For eg : if attachment feature does not exits and customer want it to be added in the middle, this can be achieved in next release

The Business analyst collects the requirement in the form of document as BRS or CRS

Business requirements specification / Customer requirement specification

- register the customer
- Get inside the app
- Create a mail and send it
- All sent mail should go to the receiver
- A copy of mail should be saved in receiver account
- Fast message service
  -
  -
  -
  -
  -
- Come out of the application

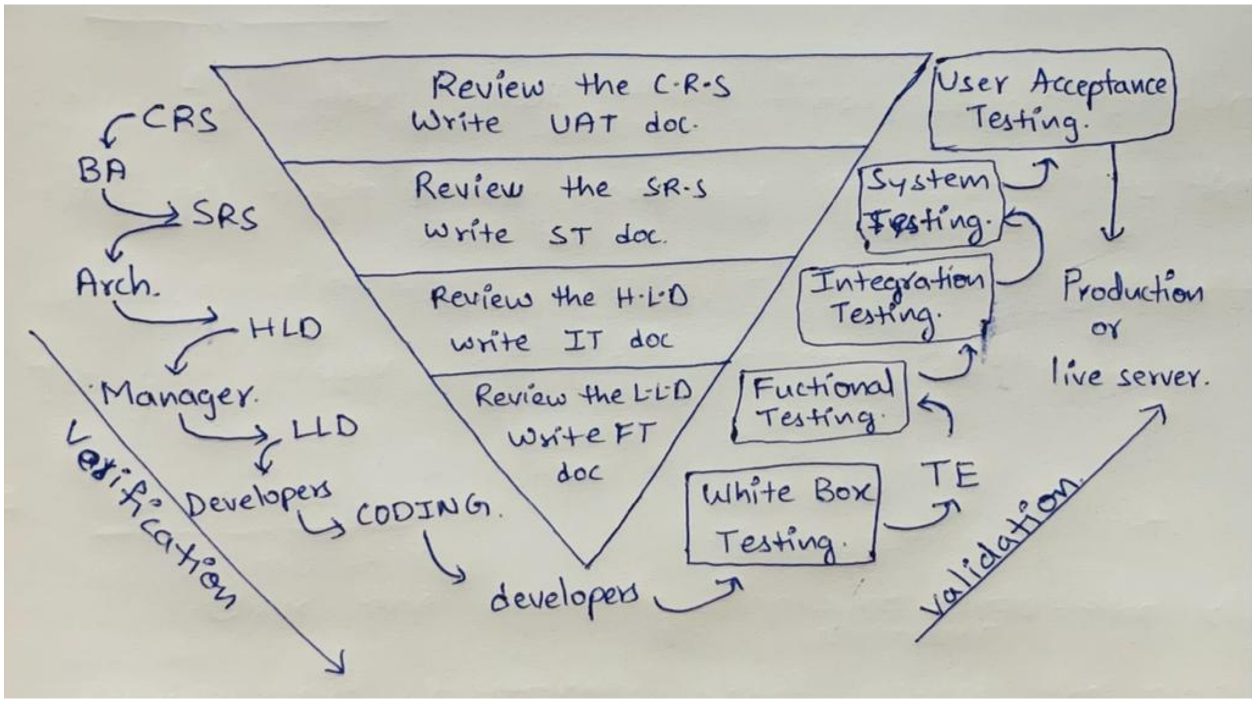
BA converts this requirement into below format so that a developer can understand it easily.

Software requirement specification / functional specification

1. Login

2.1 user name
2.1.1 user name is a text box
2.1.2. It accepts min 4 characters
2.2 password
2.2.1 password is a text box
2.2.2 it accepts min 4 digits
2.3 Ok
2.3.1 Ok is a button
2.3.2 it should be enabled
2.4. Cancel
2.4.1 cancel is a button
2.4.2 it should be enabled
2.5. Forget password
2.5.1 forget password is a link
2.5.2 it should be enabled
-
-
- Etc….!

VERIFICATION AND VALIDATION MODEL / V MODEL



We go for a verification and validation model whenever the requirements are huge and complex. In this model all the drawbacks of the waterfall model have been removed.

Verification is a documentation process and validation is an implementation process.

Advantages
- Review process exist
- Parallel deliverables exist
- No downward flow of bugs
- In between requirements changes are allowed
- TE and developer has very good product knowledge since they have hired at very initial stage

Drawbacks of V model
- If requirement changes lot of reworks need to be done
- It is a expensive approach / model
- It is a time consuming process

Quality Assurance and Quality control:

1. Assurance is the verification part whereas control is the validation part.
2. In assurance we check the right product is getting developed whereas in control we check the product which is developed according to the customers requirement.

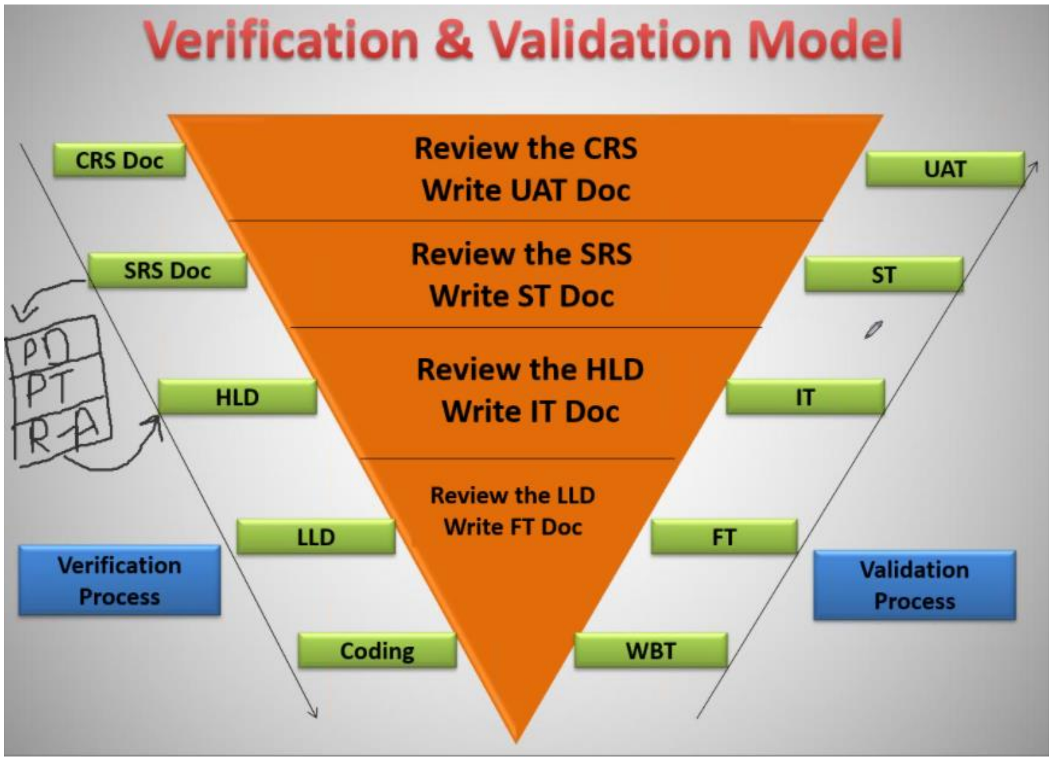
3. Assurance process starts before coding and the control process starts after coding or application is ready.
4. Assurance is process oriented i.e, in assurance we perform all the tasks in a proper process that is followed by every TE, whereas in control is product oriented.
5. Assurance is the prevention method whereas control is the detection method.
6. Assurance includes activities like go through, walk through, review, documentation, whereas control includes various types of testing such as FT , IT , ST , UAT etc…..

HYBRID MODEL

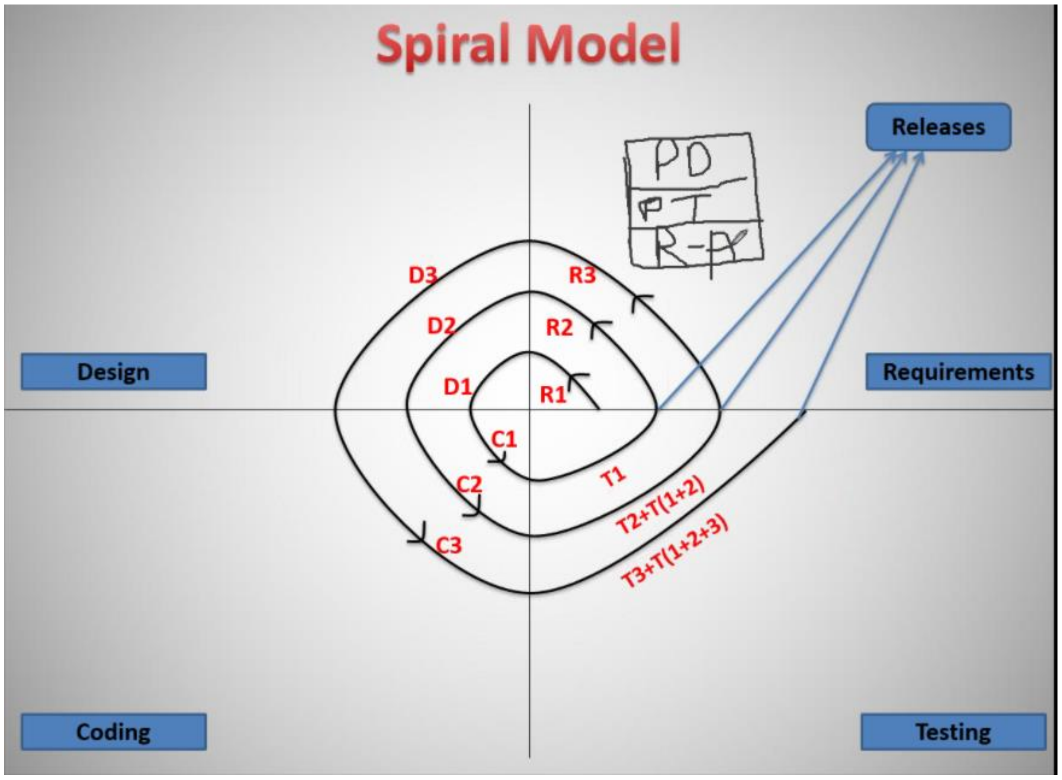
Sometimes in order to achieve the needs of the customers we have to combine the properties of two models .Such models are known as hybrid models.

The possible combination under hybrid model are

1. Prototype and V models : sometimes the customers are new to the product as well as their requirements are huge and complex. In such cases we go for the combination of prototype and V model.


Verification & Validation Model
CRS Doc
SRS Doc
HLD
LLD
Coding
Review the CRS Write UAT Doc
Review the SRS Write ST Doc
Review the HLD Write IT Doc
Review the LLD Write FT Doc
UAT
ST
IT
FT
WBT
Verification Process
Validation Process
PD
PT
R-A

2. Spiral and Prototype : whenever the modules are dependent and customers want the product at initial stage and new to the product in such case we go for the combination of spiral and prototype model.



Note :

Waterfall model cannot be a part of hybrid model since in this model in between requirement changes are not allowed.

SOFTWARE TESTING

The process of verification and validation is known as Software Testing .

Or

The process of identifying the defects and getting it fixed by the developers is known as Software Testing .

Or

The process of checking the application according to the clients requirement is known as Software Testing .

Software testing is of three types

1. White Box Testing
2. Black Box Testing
3. Grey Box Testing

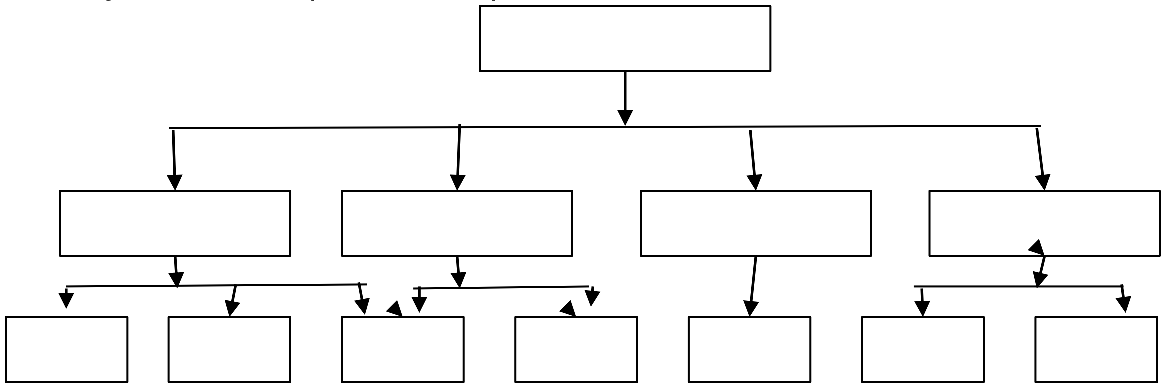
WHITE BOX TESTING

It is a type of testing performed by the developers . In this type of testing the developers check their own code line by line . It is done by the developers in order to reduce the count of errors since a small error in the code can project as the critical bug for the Test Engineers . The more the number of defects by the TE will log the more bad name for the developers .

Note :
In this type of testing the codes are visible to the developers , hence it is also known as glass box testing or transparent testing .
White box testing is not responsible to make the application stable .
Effects & bugs -- application , errors -- code .
Types of White box testing

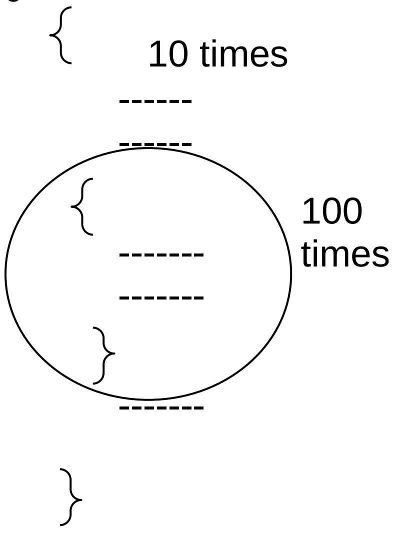
a. Path testing
b. Loop testing
c. Condition testing
d. Memory testing
e. Response testing

- **Path testing :**
  In this type of white box testing the developer checks all different flows of the application i.e, each code may have multiple branch code which further may have many sub branch code . As a developer we need to ensure that each piece of code or branch of code should get executed to perform that specific task



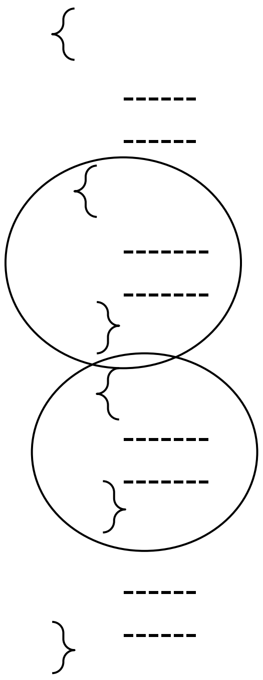
- **Loop Testing :**
  In this type of WBT the developers check for the execution of code for n number of times to derive a specific output i.e, sometimes when we execute a code the inner code may

get executed for n number of times to derive a particular output.



- **Condition Testing :**

  In this type of WBT the developer checks for the execution of the code according to condition to perform or retrieve a particular output i.e, all the lines of code written by the developer may not get executed every time , it will get executed only if the condition is satisfied.



- **Memory Testing :**
  In this type of WBT the developer ensures that the memory consumed by the code should be minimum . In order to achieve it we take help of coding standards and go for an optimal number of Lines .

- **Response Time Testing :**
  In this type of testing the developer checks for the performance of the code i.e, the time taken by the code to give a particular output . the faster the code will get executed the faster is the response time and vice versa.

GREY BOX TESTING

In this type of testing the same set of people will check the code as an application .
It is a combination of White Box testing and Black Box testing .

Note :
Grey Box testing is the bad approach , since checking the code and application by the same set of people may cause lack of accuracy .

BLACK BOX TESTING

What is Testing ?
The process of finding the defects and getting it fixed by the developer is known as testing .

The process of verification and validation is known as testing

The process of checking the application according to the clients/ customers requirements is known as Testing

The process of using the application as a user to find the defects in it , is known as Testing.

It is a type of testing performed by the Test Engineers . In this type of testing the TE checks the application according to the customers requirements .In this type of testing since the codes are not visible ,it is also known as Closed Box testing.

Note :
Black Box testing is responsible for making the application stable .

Types of BLACK BOX TESTING :

- ➔ Functional testing
- ➔ Integration testing
- ➔ System testing
- ➔ Compatibility testing
- ➔ Usability testing
- ➔ Performance testing
- ➔ User Acceptance testing (UAT)
  BA , Project Manager , Test Manager , Client perform UAT at Production Server

Software Testing

⬇

Black Box testing (by TE)

⬇

⬇

Functionality (compulsory)	Non-Functionality (customers demand)
➔ Functional testing	➔ Performance testing
➔ Integration testing	➔ Compatibility testing
➔ System testing	➔ Usability testing
➔ User Acceptance testing	➔ Ad hoc testing

Functional Testing :

(software company)

TCS

⬇

ICICI (client project)

⬇

Online Banking

⬇

➔ Login
➔ Balance enquiry
➔ Statements
➔ Amount transfer
➔ Loan
➔ Cards
➔ Payments
-
-
-
-
➔ Logout

Amount transfer (Module)

4.0	Amount Transfer
4.1	FAN
4.1.1	FAN is a text box
4.1.2	It accepts 11 digits only
4.2	TAN
4.2.1	TAN is a text box
4.2.2	It accepts 11 digit only
4.3	Amount
4.3.1	Amount is a text box
4.3.2	It accepts 50000 only
4.4	Transfer
4.4.1	It is a button
4.4.2	It should be enabled
4.5	Cancel
4.5.1	It is a button
4.5.2	It should be enabled

Inputs for FAN :

a. Valid → 11111111111 → Accepted
b. Invalid → 1111 → error message
c. More than 11 digits → 1111111111111 → error message
d. Alphabets → abc
e. Special characters → @ # → error message

f. Numeric → ab12 → error message

g. Blank → N/A → error message

Inputs for TAN :

a. Valid → 11111111111 → Accepted

b. Invalid → 1111 → error message

c. More than 11 digits → 1111111111111 → error message

d. Alphabets → abc

e. Special characters → @ # → error message

f. Numeric → ab12 → error message

g. Blank → N/A → error message

h. Same as FAN → 1111111111 → error message

Inputs for Amount :

a. Valid → 5000 → accepted

b. Invalid → 51000 → error message

c. Blank → N/A → error message

d. Alphabet → abc → error message

Functional testing :
Checking each and every module as well as components of the module individually or one by one is known as functional testing .

-Rules to be followed while performing Functional testing

1. While checking one component , inputs for all other components must be valid , so that we can locate where exactly the defect is .
2. Always start with valid input , then followed by invalid inputs ,so that we check the application in all possible ways .
3. Never check with similar types of inputs , since similar types of inputs may cause over testing which is an endless process .
4. Never assume the requirements , if the requirements are not clear , get it clarified by the concerned person .

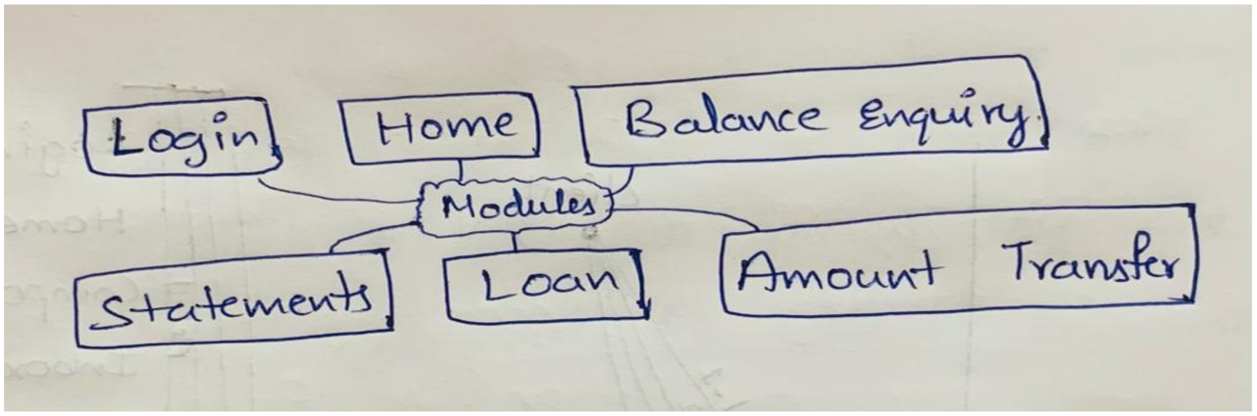
Note:
Once each and every module as well as components of the module are checked with various combinations of input we can say functional testing is over.

It is very easy to decide when to start testing but the challenging part during the software testing process is to decide when to stop testing and this decision is taken by the TE once they feel that all possible combination inputs are covered .

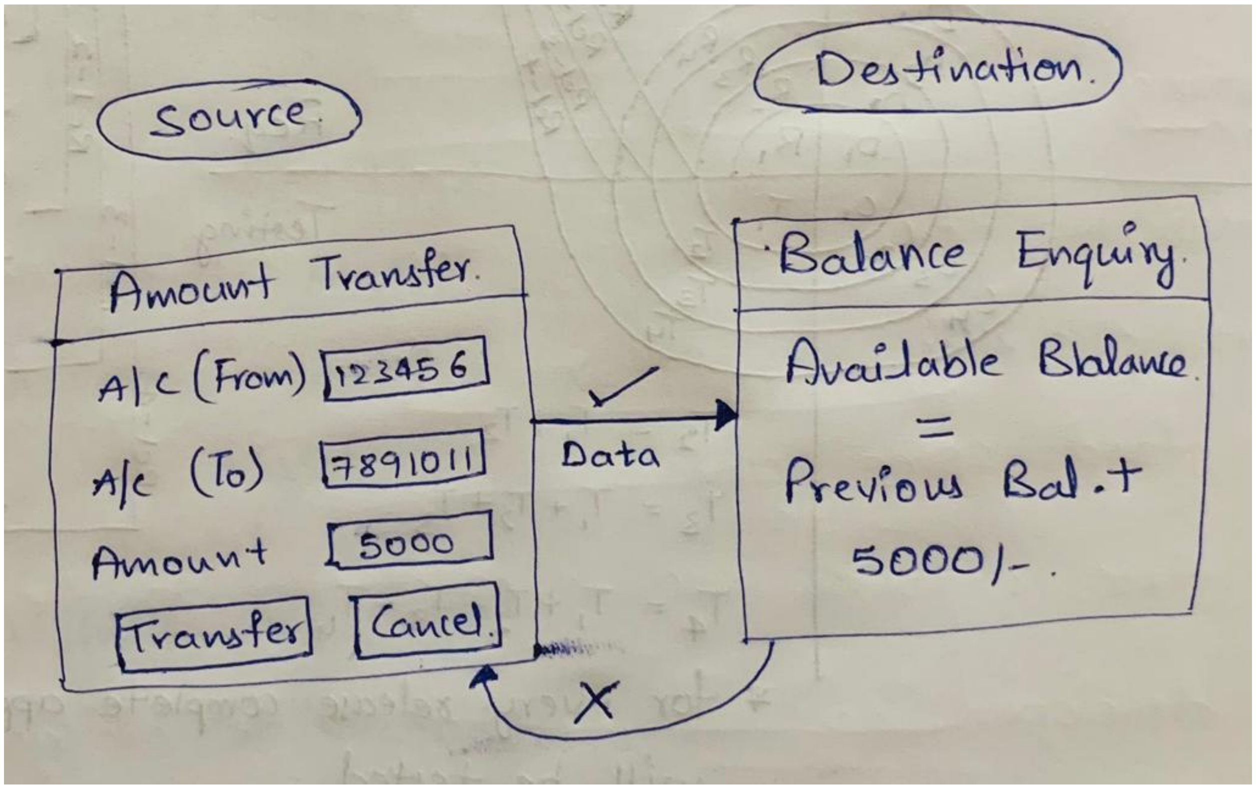
Integration Testing:

Checking the relationship or data flow between two dependent modules is known as integration testing
In order to perform interrogation testing we need to follow the following rules

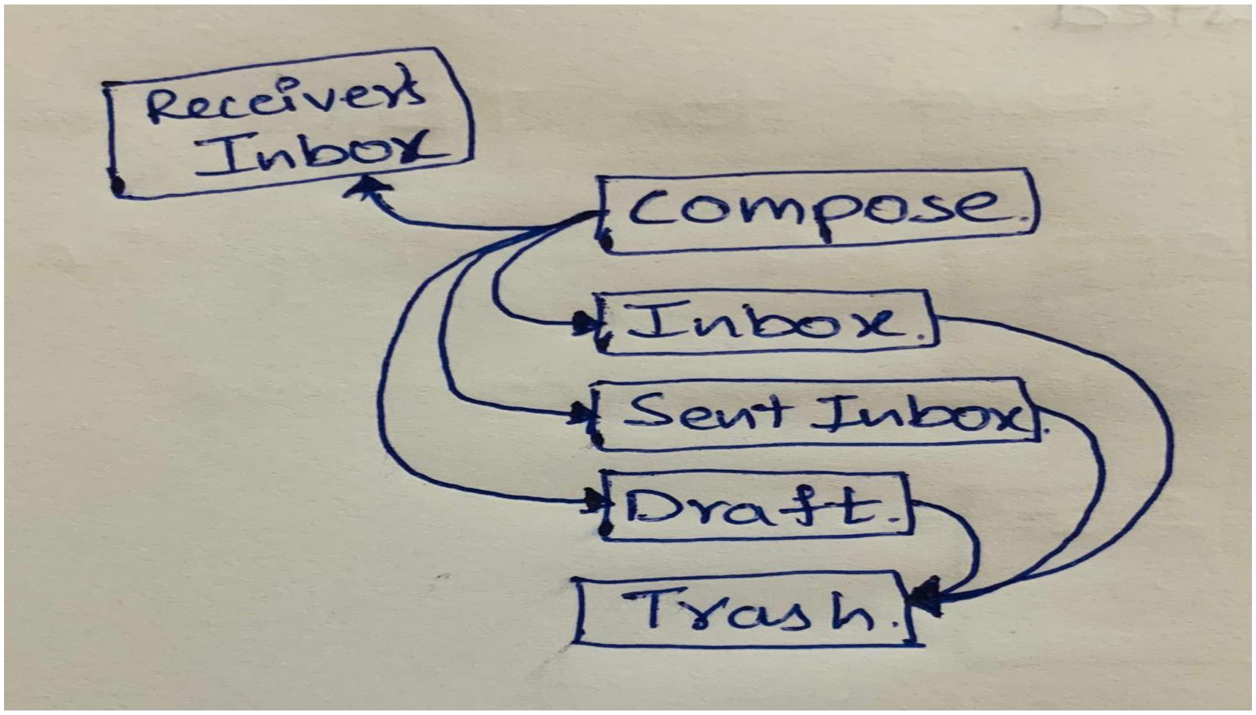
- ❖ Functional testing is done on each module as soon as the application is ready



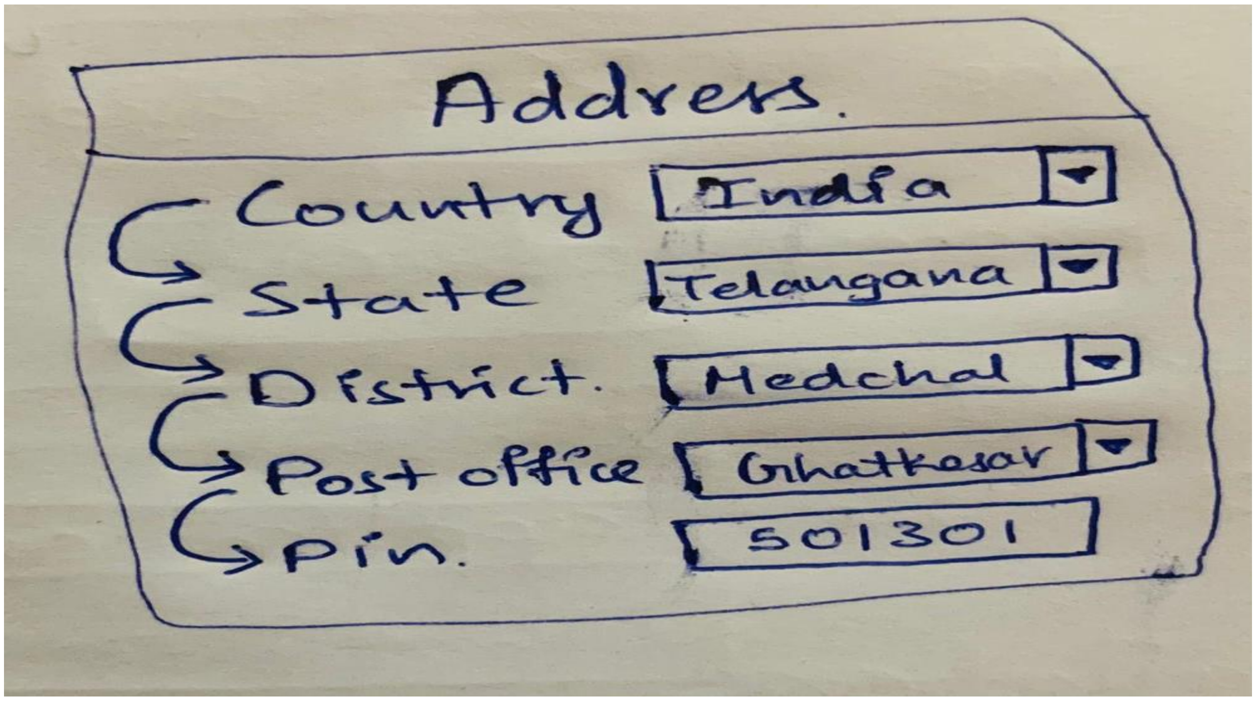
- ❖ Once FT is done we go for integration testing
- ❖ IT is done only between the source and destination. Eg: when we transfer an amount, module amount transfer is the source, after transaction e we check it through balance enquiry - destination
- ❖ In order to perform IT very good product knowledge is required . Eg:the modules are dependent on each other so we should have good knowledge about the product.
- ❖ IT is done between source to destination Eg: when we transfer an amount, module amount transfer is the source, after transaction e we check it through balance enquiry - destination.
- ❖ In order to perform IT ,we require source destination and data . Eg: the source - amount transfer and destination - balance enquiry in order to transfer the data of x amount is required.
- ❖ Data should flow from source to destination and compulsory it should get saved in the destination Eg: the amount after transferring from source to destination it should be saved in sender and receiver account.



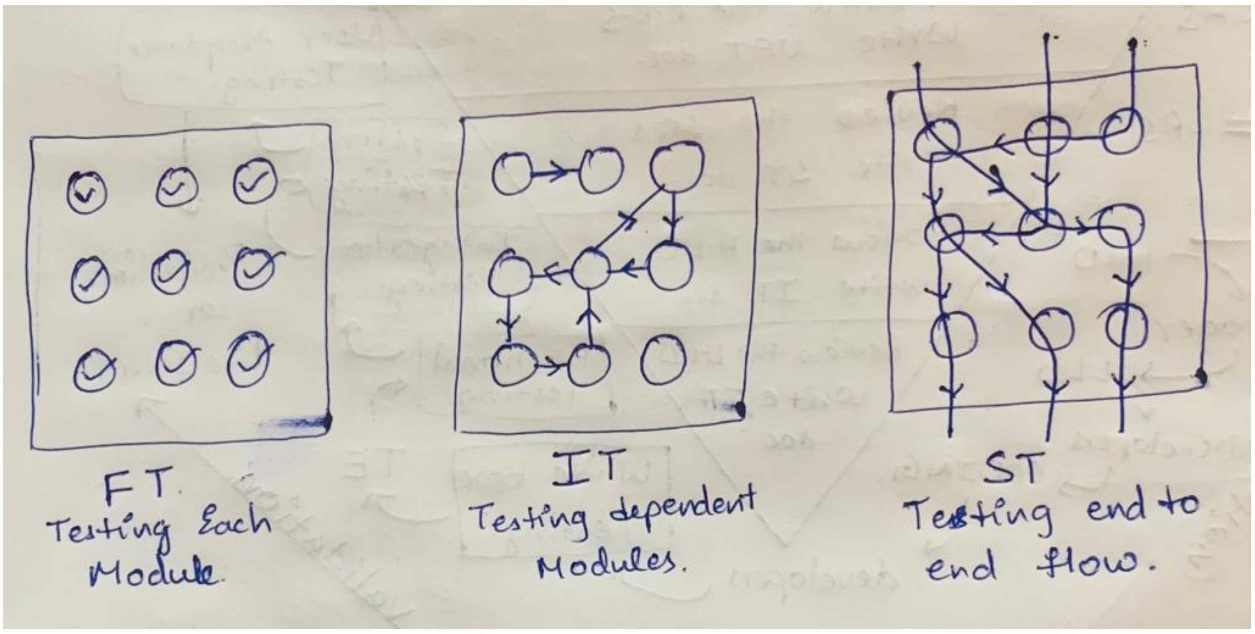
❖ Sometimes data may flow from single source to multiple destination and vice versa Eg : in Gmail when we compose a mail it should be saved in inbox, sent mail , receiver inbox, draft etc… one source multiple destination



- Sometimes data flow may happens within the module too , hence IT is done within the module



System Testing :



1. Checking the end to end flow of the application like an end user using the application is known as system testing .
2. In system testing we navigate to all necessary modules to test the application as the whole system .

-Rules

1. One application may have multiple modules ,each modules may have multiple sub modules and each sub modules may have multiple components or object
2. Every TE perform Functional Testing, Integration Testing and System Testing on their assigned model
3. All dependent modules must be assigned to same TE
4. We go for system testing once functional and integration testing is over.

-System testing on Overdraft

Situation

1. Apply for 2 months salary in advance
2. OD request will go to manager
3. Manager needs to approve
4. 15 days to credit in your account
5. Repay OD , processing charge 1000rs for one time .

Scenario

- login (employee) —> OD —> Apply OD —> Logout
- Login (manager) —> OD —> OD requests —> approve —> logout
- Login (employee) —> OD —> status —-> logout
- Login (employee) —> OD —> Repay OD —> logout

System testing is always done in a machine whose configuration is similar to that of production machine (staging server)
In order to complete system testing some time we have some waiting period , in order to overcome those waiting period we take help of SQL queries

User Acceptance Testing (UAT) :

Requirement doc
⬇
Coding then WBT (white box testing)
⬇

Functional testing	Integration testing	System testing

-Functional and integration testing will be tested in test server
-System testing is performed in staging server
⬇
User acceptance testing
-Client will test main business scenario and most frequently use scenario
⬇
Production / live server

- It is a type of testing performed by the Client before accepting the application

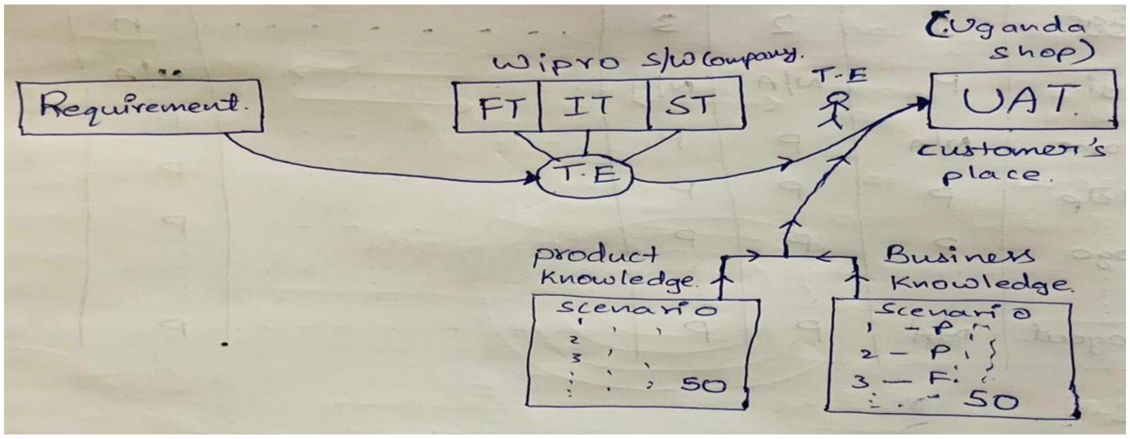
- In this type of testing the customer check the application based on their business knowledge .i.e, the main business scenario and most frequently used scenario

Difference between system testing and UAT

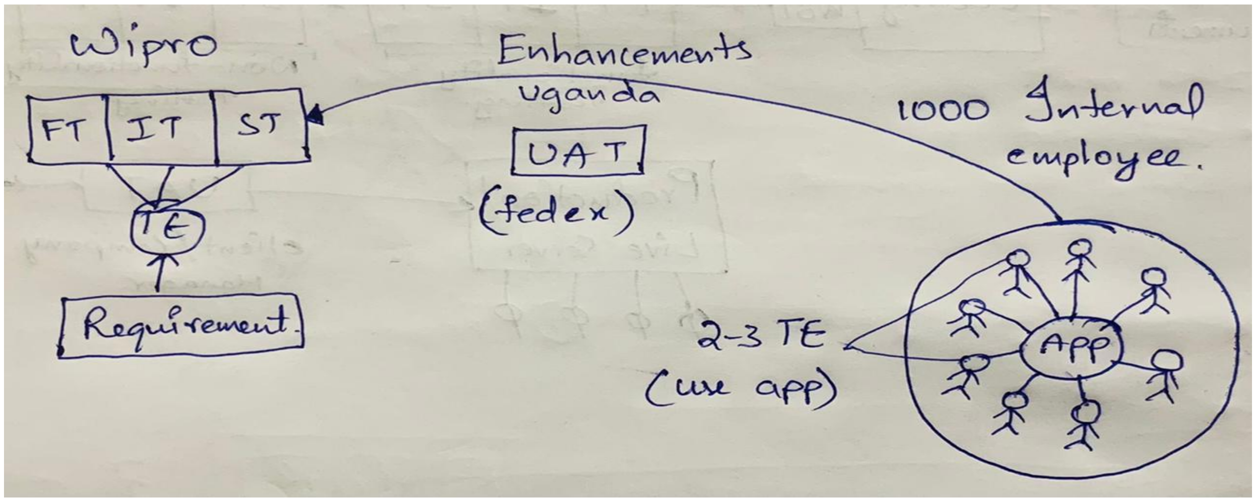
System testing	User Acceptance Testing
In system testing the test engineer will check all the end to end flow of the application in the company environment	UAT the customer will check only the main business scenario , most frequently used scenario and business critical scenario

-Cases of UAT

- ➢ UAT done by the customer (domain expert)
- ➢ UAT done by the customer (TE from the company)



- ➢ In case 2 , A TE is sent to the customers place ,because the customer cannot afford a domain expert as it is a small project as there are no regular updates
  Eg : medical shop
- ➢ TE sits with the customer for few days (say 2-3 days) to understand his business then both TE and customer prepare some scenarios based on the business knowledge
- ➢ TE again prepares another document of scenarios based on product knowledge
- ➢ Then considering both the scenarios , the TE does UATt and reports the customer about the testing status (i.e, pass or fail) , because ,customer is also having knowledge about the business
- ➢ UAT done by the customer (Internal Employee)
  Condition :
  a. Internal application
  b. Cannot be a new application



Compatibility Testing :

Once the application is functionally stable i.e, FT, IT, ST ,The application is deployed into the production or live server , where the end users start accessing the application

The end users use their own machine with various configurations to access the application as the result some of the users may Not be able to access the application.

To avoid such situations in the company we perform one round of compatibility testing.

We go for compatibility testing once the app is functionally stable

Note :

- it is not a compulsory type of testing
- We go for compatibility testing whenever we don't have control over users machine

-Types of Compatibility Testing

1. Software testing

A) Operating system : windows , mac , android , ios , unix , vista , ubuntu etc…

B) Browser : Internet ,Firefox ,Safari ,Chrome ,Etc…

2. Hardware testing

A) Size : Ram , Rom , etc ….

B) Make : Intel, Hp ,Dell, Amd, Etc….

-Operating system compatibility testing process

1. List all the desired platform Eg : windows 7, 10 , Mac, vista, Linux

2. Identify the base platform , according to the customer or the market analysis
   Eg: windows 7
3. Install the base platform in every test engineers machine
   Eg: win 7 has to be installed in all the machines
4. Every test engineers will perform functional testing, integration testing, system testing on the base platform
5. Application is stable on base platform
6. Admin will create an virtual machine server, which has installed the remaining software or operating system , and provide access to the TE
7. TE will perform end to end flow of testing
8. If compatibility issues found report to developer such as , Alignment issues , Overlapping objects , Missing objects
9. Until the application is stable in all the desired platforms.

The list of desired platform will be collected from the customer , and ask him to say the base platform or we can do the market analysis and find out the base platform , then the base platform is installed in the test engineers allotted for the project, then each TE will test the assigned module , and perform

VM server
- The admin team will install the VM server and give access to the TE
- Then the TE should search the VM server in the search box
- A tab will open regarding the search then ,select the VM server
- A new page will display with multiple boxes with multiple platforms
- Selecting the each platform a new tab opens with different OS

-Compatibility testing process across the browser

In order to check the compatibility of the application across multiple browser VM servers are not required since we can install multiple browser in single OS
But , sometimes customer want their application to be stable in multiple versions of the same browser in such case we take help of VM server

Usability Testing :

Checking the user friendliness of an application is known as usability testing, The various factors of usability testing are
- ➔ Easy to access ---   All important features must be highlighted

➔ Easy to understand --- In a simple English language
➔ Look and feel ---- Font colour, font size, background colour, background size etc…
➔ Quick response ---- It should not consume too much users time to perform an action or a task.

Usability testing is not a compulsory type of testing since there are many applications where we don't perform usability testing.
For eg: command prompt , anti viruses , and many supporting applications

It is a type of non functional testing and we go for it once the application is stable.

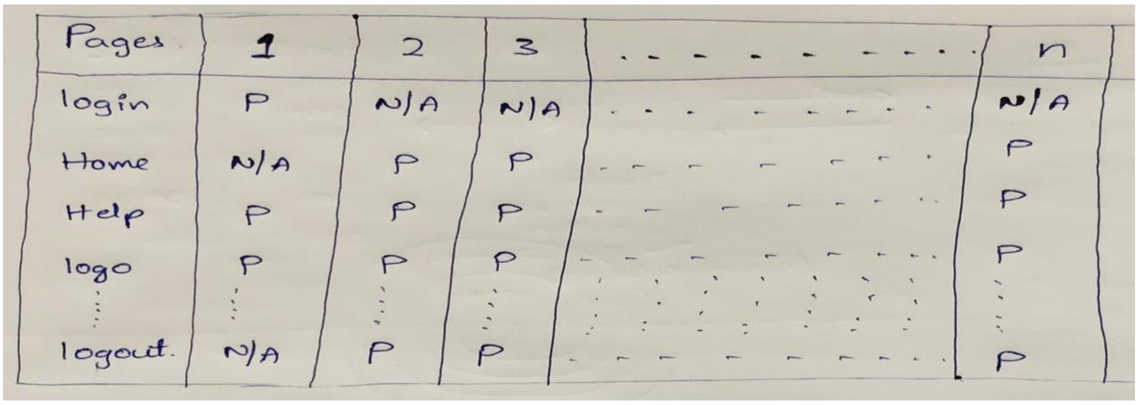
-Usability checklist

➢ application should have logo
➢ Logo must be visible in all pages
➢ Home option must be present
➢ Login button must be enabled on all pages .
  -
  -
➢ Etc…!

-Usability testing process

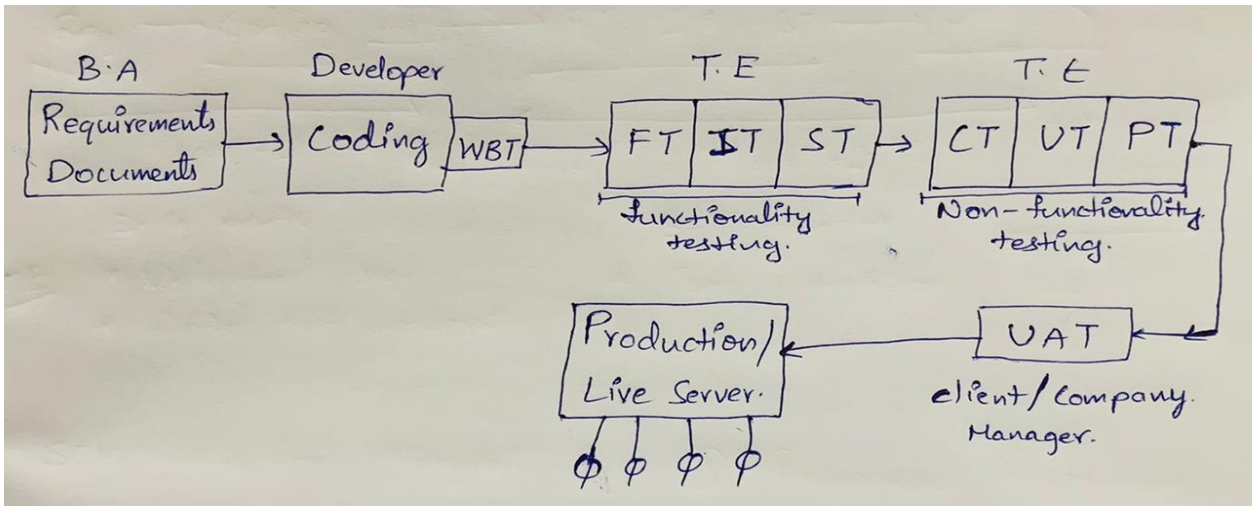
1. Identify the need of usability testing - if required then
2. Derive the usability checklist - based on the project
3. Send the checklist to customer / client for review and approval - note down the changes client required
4. Implement the changes in checklist - wait until the application is stable
5. Execute the checklist on the application
6. Derive the Usability checklist report

-Usability testing checklist Report



Pages.	1	2	3	. . . .	n
login	P	N/A	N/A	. . . .	N/A
Home	N/A	P	P	. . . .	P
Help	P	P	P	. . . .	P
logo	P	P	P	. . . .	P
⋮	⋮	⋮	⋮	⋮	⋮
logout.	N/A	P	P	. . . .	P

Performance Testing :



Checking the behaviour of the application by applying load is known as performance testing

-The various factors of Performance testing

1. Load : it talks about the num of users using the application simultaneously
2. Response time : it is the time taken by the server to give response for a particular request
3. Stability : it talks about the behaviour of the application with a load for long time without the application getting crash

Note :

1. Once the application is deployed into the production n number of users start using the application simultaneously hence , we may not have control over the number of users .
2. It is a type of nn functional testing and it is not a compulsory type of testing , we go for it whenever we do not have control over number of users using the application simultaneously
3. Performance testing is always done with the help of tools since manually checking the performance of the application is a costlier approach as well as accuracy may not exist . Some commonly used performance testing tools are
    1. J meter
    2. Load runner
    3. Neo Load
    4. Webstress Tool

-Performance testing process

- PT is always done with the help of tool and Performance TE (PTE)
- Identify the performance scenario.
   Most frequently / commonly used.
  Huge data transaction.

Most critical scenario

- Convert the scenario into scripts (core java)
- Distribute the load on scripts based on the Usage Pattern
- RUN the scripts with the help of tool
- Result.

  Max response time —2.5 sec.

  Min response time — 1.4 sec.

  Average ——- 2 sec

- Analyse the result (team) and check desired response time is equal or less than average response time
- Bottleneck - performance issues.
  hardware configuration.
  software configuration.
  network issues
- TUNNING
- Re-Run the scripts
- This process continues until we achieve the goal i.e, average response time is less than or equal to desired response time .

Scenario Eg : GMail (load - 1000)

1. Login — compose — sent box — logout (400 load)
2. Login — inbox (read / reply / forward / delete) — logout (400 load)
3. Login — chat — logout (100 load)
4. Login — settings — logout (10 load)
   -
   -
   -
   50.

-Types of Performance Testing

1. Load testing :
   Checking the behaviour of the application by applying load less than or equal to the desired load

2. Stress testing :
   Checking the behaviour of the application by applying load more than the desired load
   Eg : the customer wants 10000 responses with response time of 1.5 sec and he is getting 20000 responses with response time of 1.5 sec .

3. Stability testing :

Checking the behaviour of the application by applying load continuously for a long period of time is known as stability testing .

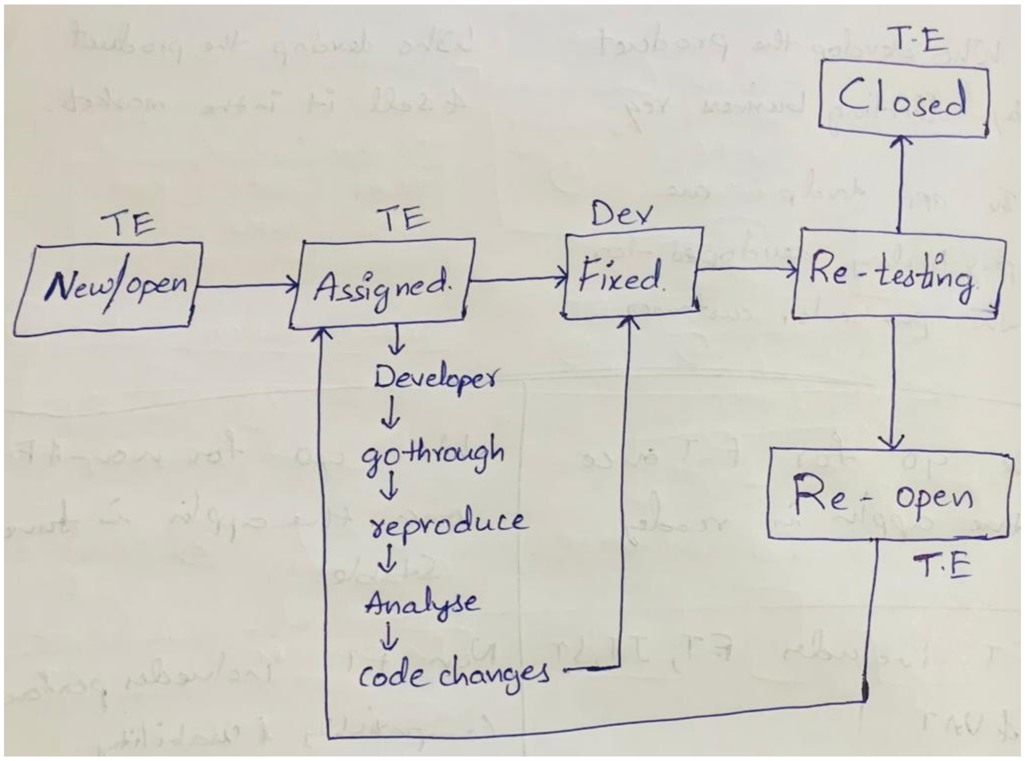
4. Scalability testing :
   Checking the behaviour of the application by increasing or decreasing the load in a particular skill is known as scalability testing
   This are two types
   a. Upward Scalability
      The response time increases as the load increases
      10000 ……… 1.5 sec
      11000 ………. 1.55 sec
      12000 ………. 1.6 sec
      13000 ………. 1.6 sec
      14000 ………. 1.8 sec
      15000 ………. 1.8 sec
   b. Downward Scalability
      The response time decreases as the load decreases
      9000 ……… 1.5 sec
      8000 ……… 1.3 sec
      7000 ……… 1.2 sec
      6000 ……… 1.1 sec
      5000 ……… 1.1 sec

Note : we should not talk about soak and volume testing until the interviewer asks us

5. Volume testing :
--Checking the behavior of the application with huge amount of data transaction is known as volume testing
--Data transaction means uploading or downloading the file
6. Soak testing :
--Checking the behaviour of the application in unfavorable environment or condition is known as soak testing

Terminology:
Error :- It is in the code , it is used by developers.
Defect :- it is in the application , it is used by TE’s.
Bug :- it is an informal name of defects , it is used by the TE’s and Dev’s.
Mistake :- it is in the documents, BA / TE’s.
Issue :- defect found by the customers.
Failure :- complete project is a failure.!

DEFECT / BUG LIFE CYCLE



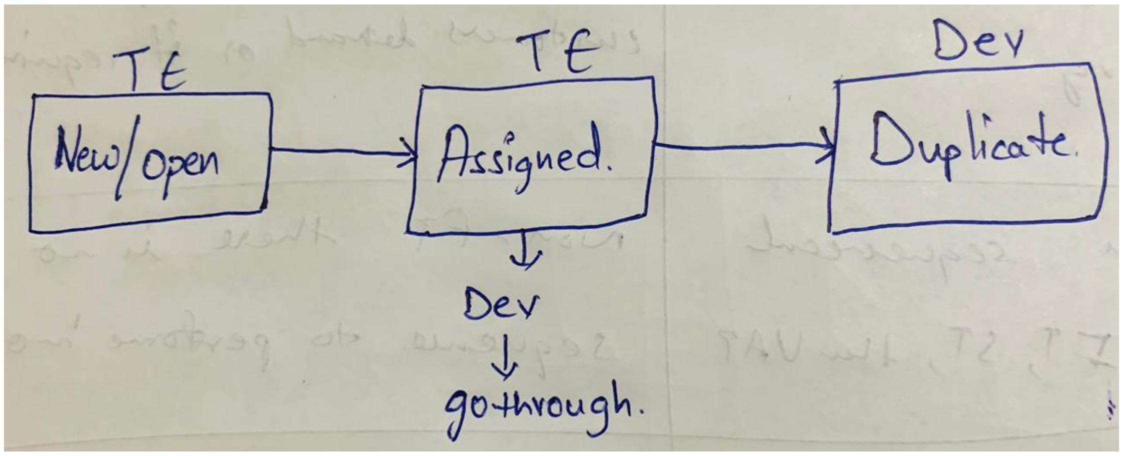
Bug life cycle talks about the complete life cycle from the time a bug is encountered until the bug is closed by the test engineer.

1. As soon as the TE encounters the bug he prepares a bug report with all the necessary information.
2. Initially the status of the bug is new or open.
3. Once the bug report is sent to the concerned developer the TE changes the status of the bug as Assigned.
4. The concerned developer goes through the bug report , reproduces the bug by solving the steps written in the bug report, analyzes where to do the code changes and finally does the code changes.
5. Once the code changes are done the developer changes the status as Fixed.
6. The TE once again retests the bug to ensure the bug is resolved or not. If the bug is no more in the application the changes the status as Closed.
7. But , if the bug still exists in such a case the TE once again changes the status as Reopen.
8. Reopen bugs are by default reassigned to the same developer; this process continues until the bug is fixed properly and closed by the TE.

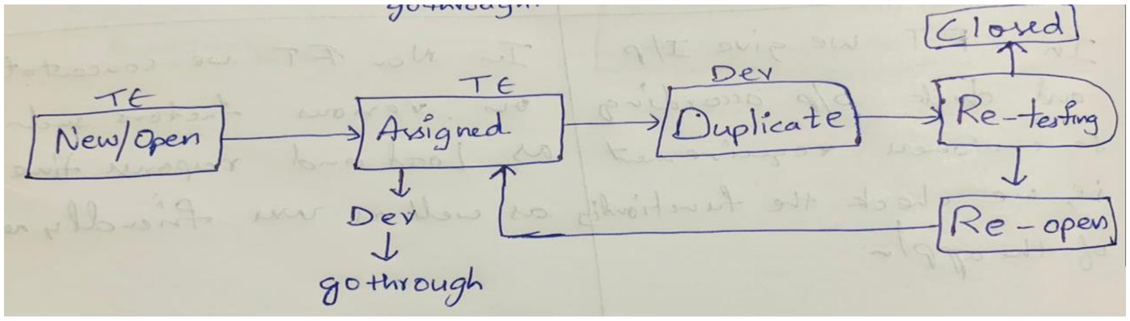
BUG REPORT
Bug ID :
Module :
Status :
Steps to Reproduce:

Other Status of Bugs :

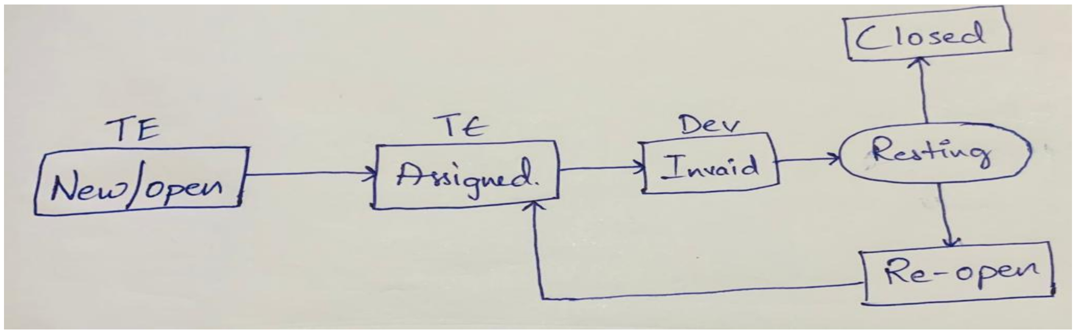
1. Duplicate bugs
   These are those bugs which are locked by multiple TE or same TE or more than one time. For the first time the TE who locked the bug is accepted and from second time onwards the statues are changed to duplicate. Duplicate bugs are not accepted by the developer.
   -Why duplicate bugs ?
   Duplicate bugs are due to multiple TE working on the same module or dependent module.
   -How to avoid duplicate bugs ?
   In order to avoid duplicate bugs as soon as the TE encounters a bug the TE should first check in the bug repository whether they find the same bug , if it doesn't exist the TE should log it to the concerned developer by keeping other TE in the loop.



   -Can we reopen the bug ?
   Yes , sometimes the bug log by one TE is not exactly the same as the previous logged by another TE. for eg : TE1 says .pdf is not getting attached whereas the other TE says .zip is not getting attached.
   If the status of the previously logged bug is closed and once again we encounter the same bug in such a case if the developer changes the status as duplicate we TE can reopen it.



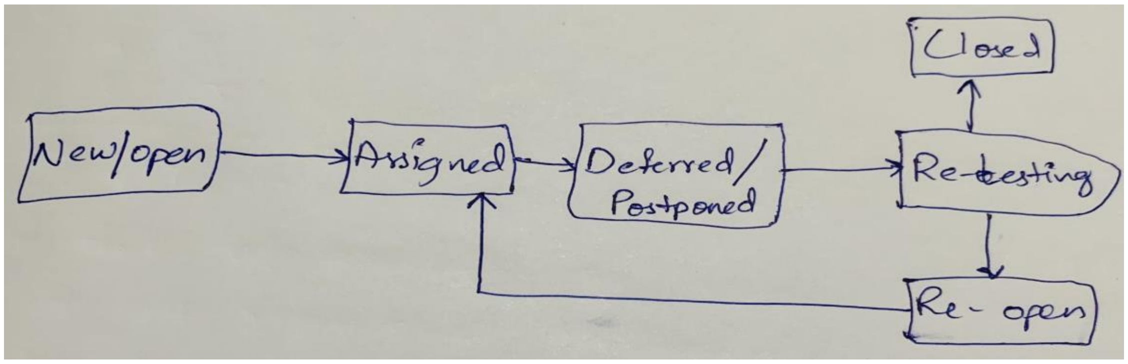
If a new bug is logged as new or open it is assigned to the concerned developer, then the developer goes through the bug report checks for the multiple logs in bug repository, if multiple logs exist he changes the status to duplicate else he fixes the bug and changes the status to fixed. The TE retests the bug whether it is solved or not, if solved the status will be closed, if not the status will be reopened and by default reassigned to the same developer, this process continues until the bug is fixed ,this is the complete life cycle of the bug.

2. Invalid bugs
   These are the bugs which are rejected by the developer i.e, the developer after going through the bug report identifies that it is not a valid bug in such a case the developer changes the status as invalid.
   -Why invalid bugs ?
   Invalid bugs are due to the misunderstanding over the requirement by the TE or by the developer.
   -How to avoid invalid bugs?
   In order to avoid invalid bugs both the developing and testing team should have very good knowledge about the requirement.
   -Can we reopen the bug?
   Yes, We can reopen the invalid bugs if the developer misunderstood the requirements eg : composed mail is getting saved in draft , it is a valid bug but developer misunderstood it, in such case we can reopen the bug.
   

   - Invalid bugs and duplicate bugs are not accepted by the developer.
   - Both of them are bad names for the TE.

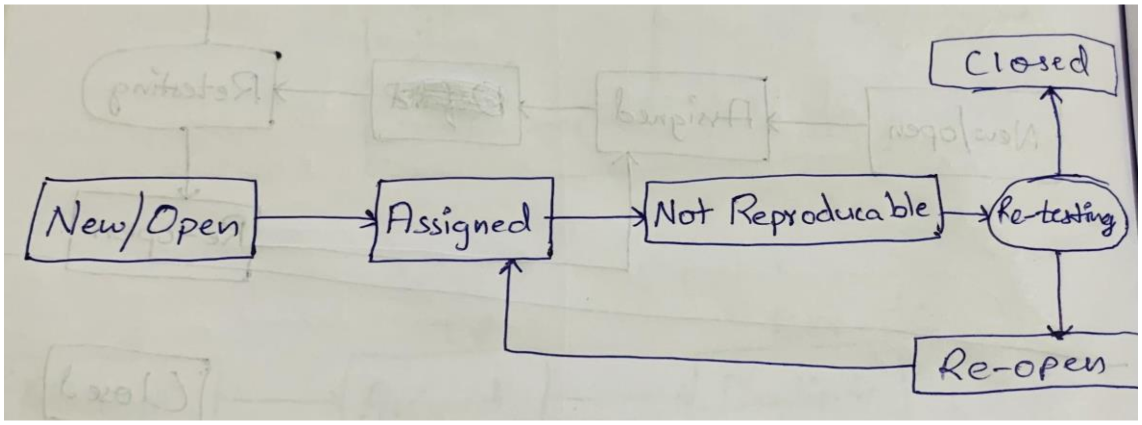
3. Deferred / postponed bug
   Deferred bugs are those bugs which are accepted by the developer but the developer postpone it to the next release.
   -Why deferred bugs?
   Deferred bugs are due to the lack of time or time constraint eg : developer had say 12 bugs to be fixed , and TE additionally give one more bug and the release date is in 3 days in such a case the bugs are postponed to next release.
   -How to avoid deferred bugs?
   Practically it is not possible to avoid deferred bugs but , as a TE, as soon as we find a bug we should report it to the developer so that the developer has enough time to fix it. We should also try to log as much as bugs at the initial stage.
   -Can we reopen deferred bugs?
   Deferred bugs can be reopened. If the bug is a major bug and has an impact in the application in such a case even though time does not permit, we have to reopen it. If a minor bug is turned into deferred in such a case too we can reopen it since the developer has enough time to fix it.

Note : All deferred bugs are minor bugs but all minor bugs are not deferred bugs.
Deferred bugs cannot be closed until it is fixed by the developer.



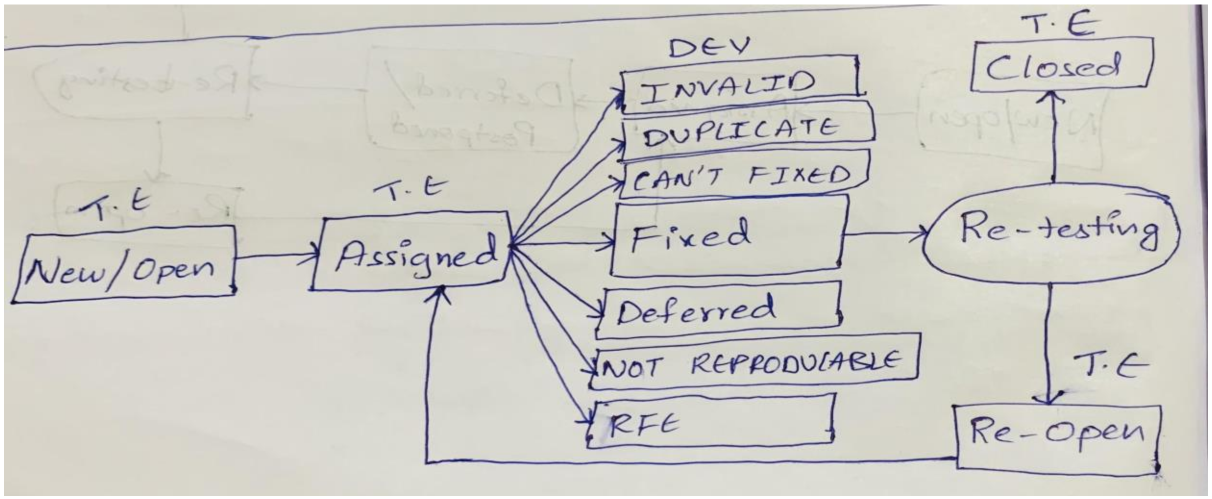
4. Can’t fixed bug
   Can’t fixed bugs are those bugs which are accepted by the developer after going through the bug report but the developers are unable to fix it.
   -Why can’t fixed bugs ?
   Can’t fixed bugs are due to the bug present in the core of the code or the architecture of the application. If these bugs are fixed the entire application may misbehave.
   -How to avoid can’t fixed bug?
   In order to avoid can’t fix bug proper architecture as well as programming language must be chosen at the initial stage.
   -Can we reopen the bug ?
   All can’t fix bugs are intimidated , if the customer is ready to live with it , we TE can change the status to be closed else we have to reopen it and this time the developer have to do the code changes.
5. Not reproducible bug
   These are the bugs which are accepted by the developer after going through the bug report but developers are unable to reproduce it by following the steps to reproduce.
   Developer will open the bug report.
   He will reproduce the bug by following the steps written in the bug report.
   He is unable to locate the same bug.
   -Why not reproducible bugs ?
   Server mismatch : TE found the bug in one server and the developer is trying to find it in another server.
   Incomplete or wrong navigation steps : TE has not mentioned .pdf is not working , but he didn't mention it in bug report , developer is checking for .xls file.
   Platform mismatch : TE found the bug in mozilla but, developer is checking in chrome .
   Data mismatch : TE has not given proper data such as wrong url, invalid credentials etc..
   Build mismatch : the bug found in previous build automatically fixed due to code changes ,now we cannot reproduce it in the current build.
   Inconsistent bug : TE is performing the test for say 100 times only 3 to 4 times he may find the bug.

-How to avoid these bugs ?
In order to avoid not reproducible bugs the TE should prepare a proper bug report with all the necessary information along with the screenshot of the bug.
-Can we reopen it ?
All Not reproducible bugs are valid bugs hence, we TE will once again do the changes in the bug report and send it back to the developer.



6. RFE (Request for Enhancement)
   RFE's are the suggestions given by the TE for the better performance of the application, technically it is not a bug. Since the customer has not asked for it the developer has not developed it but as a TE we need to inform it to the developer since the consequence of it may impact the user in order to fix it the developer have to do the code changes and again we have to retest it , hence we keep it in other status of bug life cycle.
   All RFE's bugs are informed to the customer along with their consequences , if the customer wants the feature to be implemented , the TE will reopen it else the TE will change the status as closed.

FINAL DIAGRAM OF BUG LIFE CYCLE



The bug life cycle talks about the complete life cycle of one bug.

Bug Report Template

Refer sheets.
Note :
Logging and tracking of bugs in the company are always done with the help of bug tracking tools; some commonly used bug tracking tools are jira , bugzilla, mantish , QC/ALM.
Advantages of Bug Tracking Tools :

1. It saves the time of logging the bugs.
2. Easy to track the status of the bugs.
3. It gives a centralized storage(repository).
4. It is more secure.
5. No human error.

Severity:
Severity talks about the impact of the bug on the application i.e, how much the bug is affecting its application is known as its severity.
Severity is decided by the TE but it can be changed by the developer since wrong severity affects the performance of the developer.
The diff labels of severity are Blocker / Showstopper , Critical , Major , Minor.

- Blocker / Showstopper Bugs
  Blocker bugs are those bugs which stop the entire testing process.
  Comose is not working on a blank page , effects SI , Inbox etc… - blocker bug.
- Critical Bugs
  Critical bugs are those bugs which are found in the main functionality of the application.
  SI is not working -- critical bug.
- Major Bugs
  Whether supporting functionality or main functionality is not working it is known as a Major bug.
  Pdf file is not getting attached - Major bug.
- Minor Bugs
  Minor bugs are those bugs which are found in the UI of the application.
  Font size , color , style - Minor bug.

Priority:
It talks about the sequence in which the bug needs to be fixed i.e, when which bug needs to be fixed by the developer is decided by its priority.
Priority is decided by the TE and it cannot be changed by the Developer since developers get a chance to change the priority. They will fix all minor bugs first in order to reduce the count of bugs. If all major bugs are fixed later the TE will not have enough time to check the impact area.
The different labels.

- Urjent bugs
  These are those bugs which need to be fixed in the next immediate build i.e., bugs found in build no 10 should be fixed in build no 11.
- High

These are those bugs which can be fixed in the upcoming 2 or 3 Builds i.e, bugs in build 10 can be fixed in 12 0r 13.

- Medium
  These are these bugs which can be fixed in the upcoming few builds or intermediate builds.
- Low
  These are those bugs which can be fixed in the last 2 0r 3 builds.

Severity and Priority of Bugs :

1. High Severity and High Priority.
   Composing mail , couldn't be seen in the receiver's inbox.
   Unable to login with valid credentials.
   Ordered a mac , but received a dell.

Note : All the blocker bugs.

2. Low Severity and Low Priority.
   Background colour should be creamish white, but it is full white.
   Size of the logo.

Note : All UI related bugs.

3. High Severity and Low Priority.
   Compose butting is getting disabled after sending 200 mail, we need to relogin.
   Help module is not working.
   Products added in cart get deleted automatically after adding 25 products.
   Autoplay of video is not working after playing 20 videos.
   After an hour of call it automatically gets disconnected.

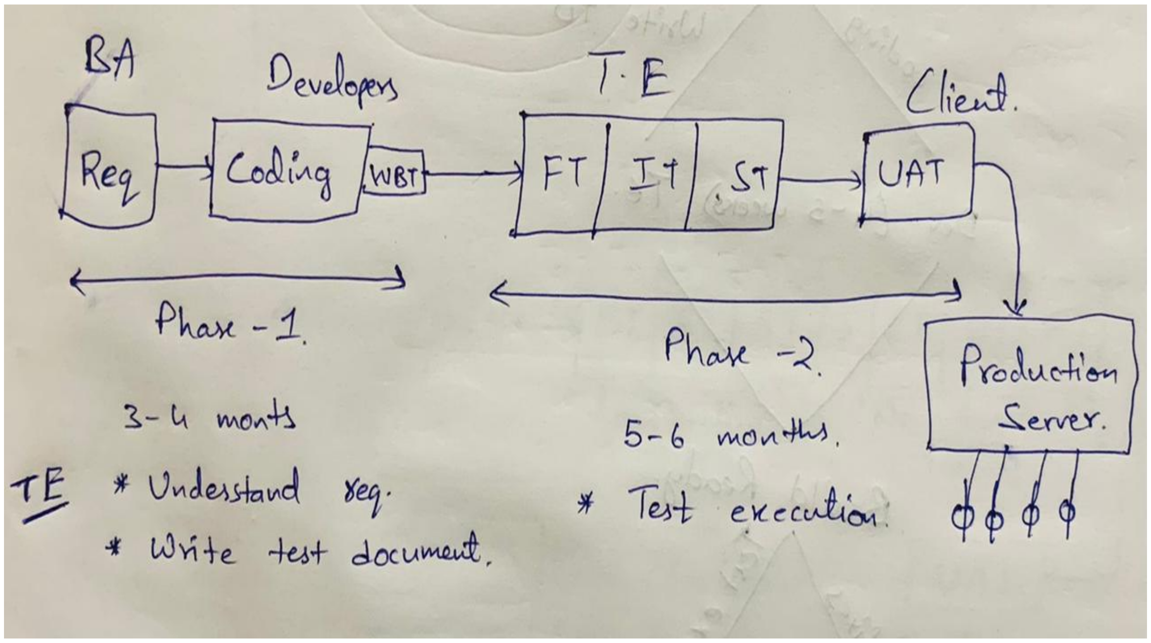
Note : Important individual module , since it has no impact on main functionality working for an extent.

4. Low Severity and High Priority.
   Colour of the logo.
   Background colour and text colour are the same.
   Font style is not readable.
   Autoplay of preview is not working.
   If we logout from the application it takes 5 min.

Note : We cannot use the application because of its appearance and any moral issues.

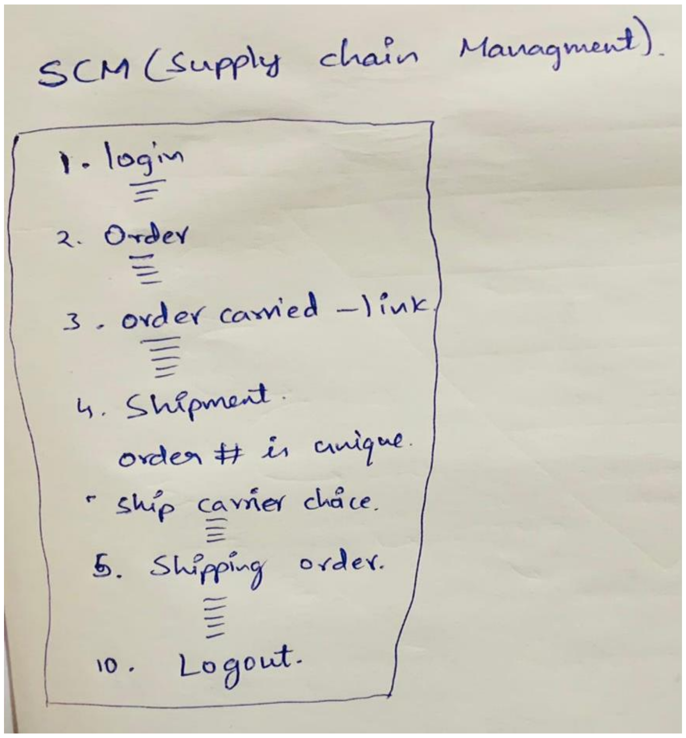
Questions:
Explain bug life cycle ?
Explain other status of Bug -- why / what / how to avoid / life cycle ?
Explain bug report template ?
Advantages of tools ?
Explain severity and priority of bugs how to decide it ?
Difference between severity and priority with examples ?

TEST DOCUMENTS



-What is a test document ?

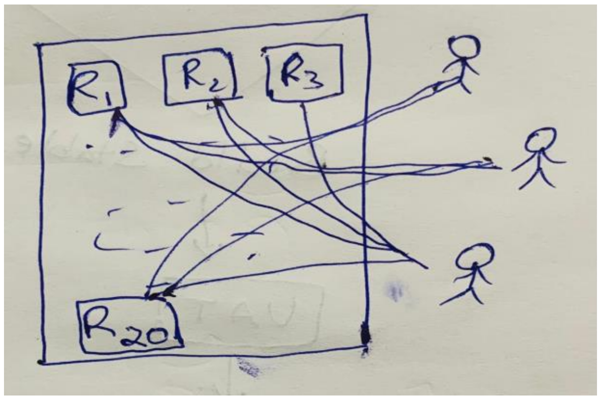
- ➔ It is a reference document which is prepared by the TE when the developer is completely busy in writing the code.
- ➔ Generally , we write this document , so that we can refer to them for future test execution (whenever we are testing the app).
- ➔ Once the test documents are ready the entire execution depends on this document itself.
- ➔ Test documents are also the backbone of the software testing process.


SCM (Supply chain Managment).

1. login
2. Order
3. order carried - link
4. Shipment.
   order # is unique.
   - ship carrier choice.
5. Shipping order.
10. Logout.

Whenever we get the requirement we should understand
-terminology
-relationship between modules
-functionality

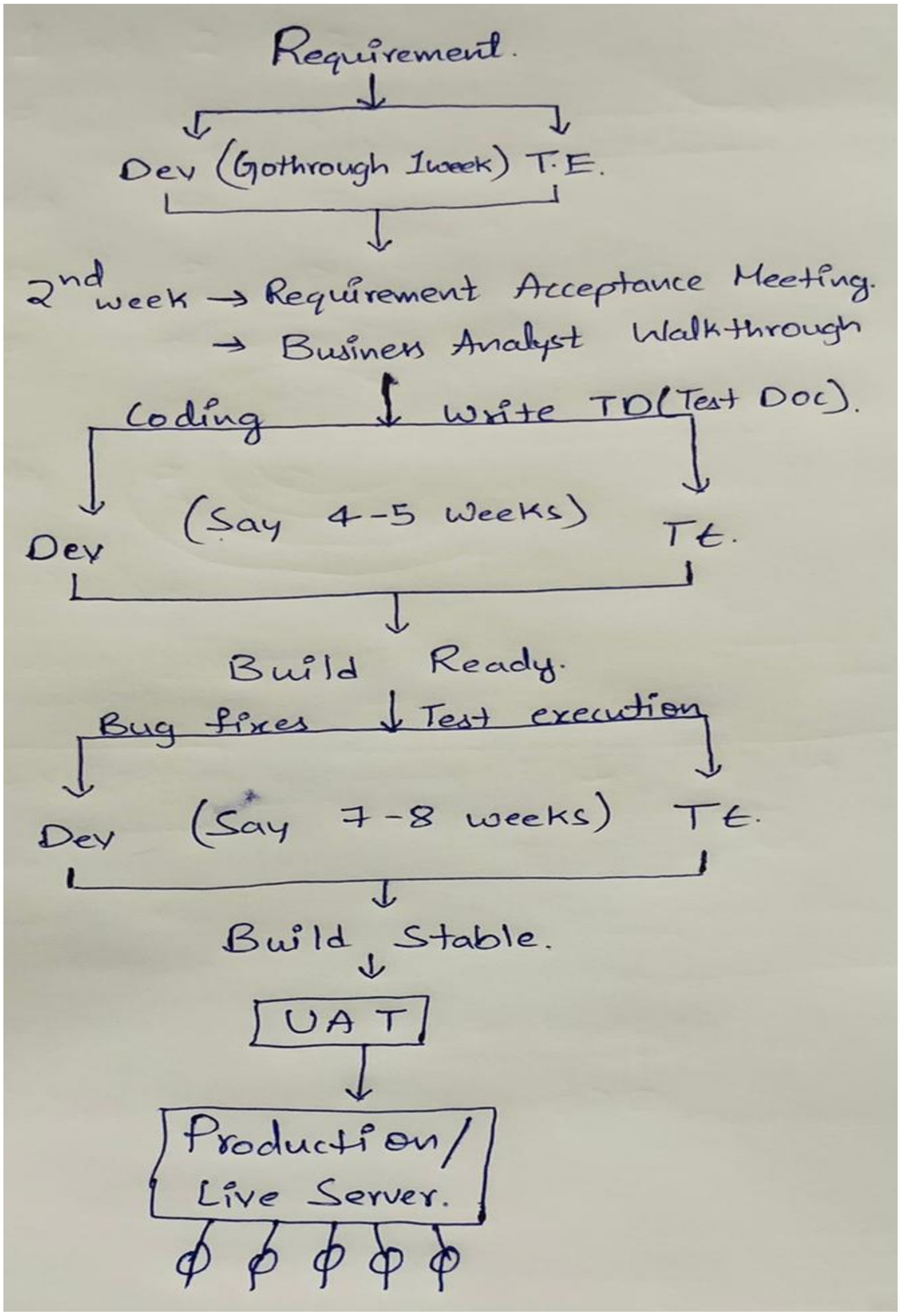
1. Requirement Repository :

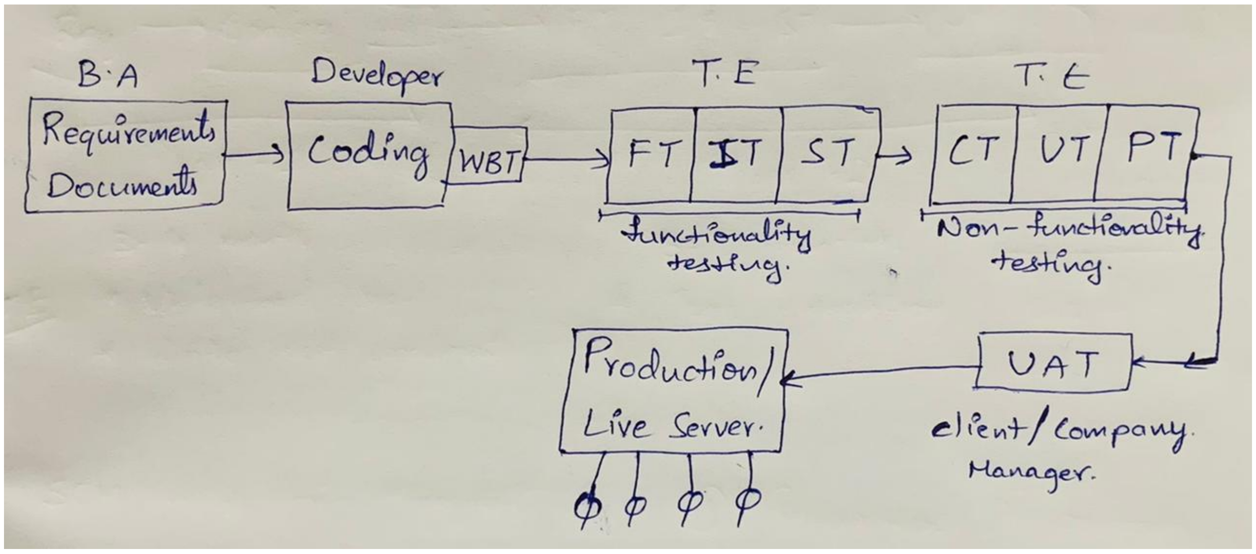


Requirement repository means ,where all the requirements are stored in one place .

2. Assigned Modules :

The modules are assigned to TE by TL (Team lead / Test lead)





Considering the project to be of 1 year , during the 1st phase of product development, which is of time duration 2-3 months the developer does coding and in the 2nd phase, which OD of time duration 4-5 months, he does bug fixing .

During the 1st phase, the TE understands the requirement as well as writes test documents , in the 2nd phase he does the test execution

Drawbacks of TE understanding the requirements and keeping in his mind only :

- The TE may Forget test scenarios
- No testing consistency
- Flow may not remain the same when testing different releases
- The no of scenario may differ from Release by release
- If one TE leaves in 1st phase , before starting 2nd phase , the new TE hired would need to do 1st phase once again so it is time consuming

What is present in the test documents ?

1. The different scenarios that will be used for testing
2. Navigational steps - it means the way of moving from one module to another , and their relationship
3. The different inputs that would be used for testing

Test documents are of two types

1. Test scenarios
2. Test cases

TEST SCENARIO

This is the high level document which talks about all the possible situations or multiple ways of testing the application

Generally , we write the test scenario so that we can understand the complete flow of the application

Note: scenario doesn’t consist of navigation steps and inputs hence it cannot be used for test execution process

-Rules to be followed to prepare test scenario :

- ➔ It should be in simple English language and not more than one or two lines (clarity exists)
- ➔ It should be in proper sequence, i.e, module by module so that we should not miss any scenario
- ➔ We should mostly talk about module level but in some cases we may talk about components
- ➔ While performing testing we TE create some data, hence DELETE should be last action
- ➔ It should be always written in DO and CHECK format , i.e , we perform some action and check for the result

Test scenario on Gmail Application :

Modules -- login, compose, inbox , send items , draft, trash, spam, chat, help, settings, logout .

Login :

1. Enter the valid URL, Login page must be displayed
2. Enter the valid credentials , home page should be displayed
3. Enter the invalid credentials , error message must be displayed
4. Logout of the application, login page must be displayed
5. Enter the credentials and refresh the page, the login page must be refreshed.

Compose :

1. If we click compose, the window / tab should be displayed .
2. Compose a mail with valid details, should be displayed in receiver inbox
3. Whenever you compose a mail, a copy of mail should be saved in sent items
4. Compose a mail to yourself, it should be displayed in your inbox

5. Compose a mail with invalid details, should be saved in draft

Inbox :
1. When you go into the inbox , all messages /mails must be displayed .
2. When you click on mail/ message , you should be able to read it .
3. The read message can be replied / forwarded , it must be displayed in receiver inbox
4. Save the message in the inbox , it should be seen in draft .
5. If we delete the message in inbox , it should be seen in trash

Send Items :
1. When you go into sent items , a copy of all sent mails must be displayed .
2. When you click on the mail in SI , it should able to read , reply and forward
3. The copy of a saved mail in SI , should be able to modify ,i.e, add or delete data.
4. Sending a mail from SI , must be displayed in receivers inbox
5. The copy of mail deleted in SI , should be displayed in trash

Draft :
1. Go to draft , all saved mails should be displayed .
2.

Trash

Spam
Chat

Help

Settings
Logout

TEST CASES

It is an In-declared document , which talks about step by step procedure to test an application .

The test case consists of all the navigational steps and inputs as well as the scenarios which need to be tested for the application .

Generally , these cases once written are followed by all the test engineers for the test execution process .

Note :

In order to write the test cases we should have following things

- Requirements -- based on which we derive all the inputs and navigation steps.
- Test scenario -- this makes us understand the complete flow of the application.
- Test cases format -- so that every TE follows the same approach of writing the test cases, which maintain the uniformity .

Test Scenario	Test Cases
TS are the high level document which talks about all the possible situation or flow or ways to use / test the application	TC are the in-detailed document which talks about step by step process to test an application
TS are used for understanding the flows of the application	TC are used for performing test execution process
TS does not consist of input parameters and navigation steps	TC consists of input parameters and navigation steps
In order to write scenarios we don't follow any template	In order to write test cases we follow some template

Test Cases Template :

-Header

Test case Name / ID	

Test case type	FT / IT / ST
Requirement	
Module	
Status	Pass / Fail
Severity	Critical / Major / Minor
Release	
Version	
** Pre condition	
** Test data	
Summary	

Body

Steps	Description	Inputs	Expected results	Actual results	Status	Comments

Footer

Date	
Author	
Reviewed by	
Approved by	

-Pre condition

Pre condition is the necessary condition which needs to be fulfilled in order to execute a particular test case.

These conditions are fulfilled by the particular test engineer working on that particular assigned module.

Pre conditions vary from test cases to test cases.

In pre-condition the test engineer creates the data. But , datas is not pre condition.

Inputs cannot be pre condition.

Action cannot be pre condition.

Test data :

Test datas is the valid input which is created by the test engineer for that particular test case.

These datas vary from test cases to test cases.

Summary :

Summaries means one line description for that particular test case.

-Definitions

Test case Name / ID : every test case has its unique name .

Test case Type : Test case type written by the TE's FT / IT / ST .

Requirement : for every module we write multiple test cases

For eg : login has 4 test cases , compose has 10 test cases .

Status : Pass / Fail .

Severity : how much it is important or impactful . It is of three types Critical / Major / Minor .

Release : one release may have multiple versions .

Pre Condition : In order to perform one action by TE’s

For eg : valid login registered account must be created.

Test case template login module excel sheet.

Functional testing test cases excel sheet.
Integration testing test case excel sheet.
System testing test case excel sheet.

Test Case Design Technique :

- ❖ These are the rules or methods or procedures which need to be followed by every test engineer while writing the test cases , so that we achieve the maximum test coverage.
- ❖ When we follow these techniques the test cases become process oriented rather than person oriented .
- ❖ The various test case design technique are
    1. Error Guessing
    2. Equivalent Partitioning
    3. Boundary Value Analysis

-Error Guessing

In this technique or method every test engineer starts Guessing the value based on their understanding the requirements .

It has no rules and it is purely person oriented .

Eg : 3000 to 5000

3100	4200	4700	A1#$@
3500	4400	4900	12ab
3800	4500		ABCDE

-- Disadvantage Of Error Guessing :

1. It is person Oriented .
2. Minimum Test coverage may not be achieved .
3. Some important inputs may not be covered .

-- Note :

Error guessing technique is applicable for all the types of components that is in every component input by default , we have to go for Error Guessing

-Equivalent Partitioning

- When every input is in the range of value . we, check for one valid and two invaid inputs
- Whenever the input is the List of values / Set of Values , we need to check for one valid and two invalid inputs .
- Whenever the input is a Boolean we check for both true as well as false condition

Disadvantage of Equivalent Partitioning :

1. It is not applicable for the range of values since we are not covering the important input .
2. This technique is process oriented
3. It will check for the minimum and maximum test coverage .

-Boundary Value Analysis

Whenever the input is the range of value we all always go for Boundary Value analysis, that is if input is a to b then a-1 , a , a+1 , b-1 , b, b+ 1 .

Eg : 3000 to 5000

2999, 3000, 3001, 4999, 5000, 5001

Note : whenever the input is the set of values , we always go for the equivalent partitioning , But in order to achieve maximum test coverage for the set of values (drop down) we have to test for all the values present in the list .

1. Range of values -- text box (1 to 10)
2. Set of values -- List of values
3. Boolean -- Radio button

Error Guessing : It is applicable for all the types of components

Equivalent Partitioning : It is applicable for the set of values and boolean

Boundary Value Analysis : it is applicable for range of Values

Eg :

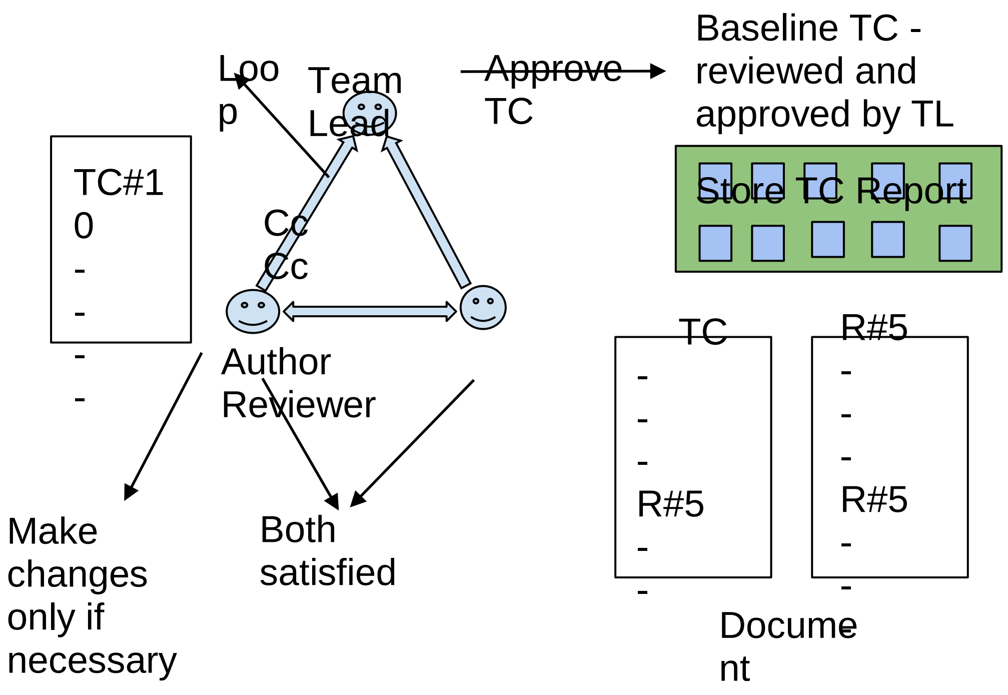
1. Username :___________ ----------------BVA (aa, abc, venu, shiva , safari)
   - it accepts 3 to 5 characters
2. Password : ____________ --------------BVA (99,- 000, 001, 998, 999, -1000)
   - it accepts 3 digits only
3. Gender ☐female ☐male --------------EP (true / false)
4. State :_____________☐ ---------------EP (all states)
   - All states of India
5. DOB :_____________☐----------------EP / BVA / EG
6. Amount : _____________ ---------------BVA (9999,- 10000, 10001, 19999, 20000, -20001)
   - 10000 to 20000

Test Case Review Process :

Once all the test documents are written and before starting the test execution process , we must have one round of review and approval powers , so that - eerie TE's will be serious about writing scenarios / inputs / navigation steps .

Note :

1. Every TE's are the author as well as reviewer
2. We send the TC's for review process only after all the TC's are written completely
    - Because , the other TE (when we are sending for review) is busy writing his TC .



- ➔ First the TE will write the TC based on the requirement
- ➔ Before sending it for reviewing he will first go through the TC to find any mistake
- ➔ He will send the TC to the reviewer through a mail
- ➔ The reviewer will go through the TC and find the requirement number which is saved in the requirement repository
- ➔ He then understands the requirement for say ½ hr .
- ➔ He then starts reviewing the TC based on his understanding of the requirement and finds out the mistakes
- ➔ If he finds any mistakes , he will prepare a review document specifying all the mistakes along with review comments
- ➔ The reviewer will then send ths document to the author through a mail
- ➔ The author will g through the review document and make changes only if necessary
- ➔ The author will again send it to the reviewer for review and this process continues until both the author and reviewer are satisfied . After the reviewer is satisfied he will send it to the TL.

- ➔ During this entire process , the TLis also kept in a loop , in other words , the TL is also sent the mail , both TC as well as reviewed documents .
- ➔ The TL will review the critical TC’s or the TC’s ehre the review comments are not proper
- ➔ After that the TL approves the TC
- ➔ Then after the TC has been written, reviewed and approved the TL saves the TC in the TC repository .

Mistakes :

1. Follow not proper
2. Coverage not achieved
3. Expected result not correct

Test case Review Template :

- Refer to the excel sheet .

- Review Ethics (Rules for the Reviewer)

1. Check for the maximum coverage and proper flow .
    - All the inputs and the scenarios must be covered as well as the navigation steps should be checked .
2. Find mistakes , not the solution .
    - Author will be serious while writing the TC’s .
3. **** Both author and reviewer are responsible for the TC.
    - Content of the TC (any missing scenario or any other problem) both are responsible .
4. Look for the content , not the author .
    - Eg : say if the author is a girl and the reviewer is a boy , then if there exists any mistake in TC the reviewer must raise it .

SOFTWARE TEST LIFE CYCLE

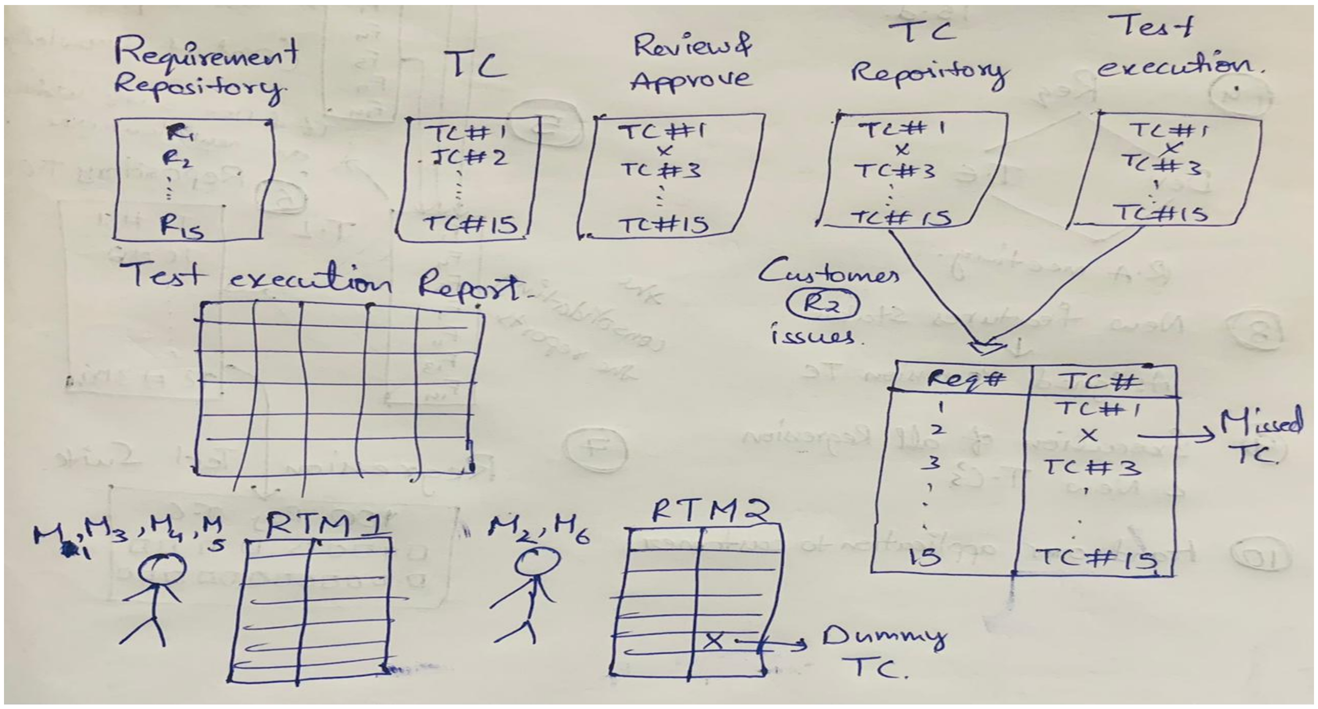
There are two types

1. RTM
2. Test execution reports.

Requirement Traceability Matrix / Cross Reference Matrix :

- As soon as we get the requirements it is stored on the requirement repository
- The TE's wil gothroughthe requirement and start preparing the test cases bose on the requirement
- Suppose , while writing the test cases for req#2 , the TE found difficulty in writing the TC (difficulty in the sense he understood the terminologies , functionalities and dependency but he was unable to understand the approach i.e, the navigation stes and c/p's that would be required for testing)
- So , he decided to skip / postpone the test case and continued writing the TC's for other requirements .
- But , at the end he forgot to write that skip tc and sen for the review and approval .
- During review , the reviewer TE was not able to track the missing TC , because the job of the reviewer is to review only those TC's which he receives via mail to review .
- During approval the TL was also not able to track the missing TC , because he has to approve so many TC's , so he can't open each and every TC and track if any TC is missing , so he only reviews only basic and critical TC's and approves them
- This TC's as soon as the TL approves , he saves it in the TC repository .
- Next test execution is done . during test execution als the TE`s is not able to track the missing TC because application is executed based on the TC . in other words TE , first opens the TC and then the application , not he other way around he will open a TC , and execute it if at all there is any depending i.e, data flow , he will only check of dataflow is proper or no and come back in the previous module , instead of
- Then test execution report is prepared based n the status of TC , during preparation of TER also , the missing TC is not tracked
- Next it is moved to the production and when the end users start using the application they find lot of issues related to req#2
- This is a problem which is caused due to the fault of the TE , so this is a bad name for the TE as well as the company
- So before the test execution process or before executing the TC's , RTM is prepared .
- Here , each TE will prepare their own RTM according to their assigned module , this RTm consists of 2 columns , req# and test case name i.e, mapping the tet case name with the particular requirement .
- All the TE's send their RTm document to the TL, and TL consolidates the RTM's and prepare a single consolidated RTm
- But before preparing the consolidated RTm he will cross check whether all the TC's names that are specified in the individual RTm documents actually exist in the TC repository or not .
- If there is any dummy TC name , he can easily find it by that method (of preparing RTM document) the TL can easily track any missing TC
- This process of preparing RTM is done before test execution , because we have enough time to write the missing TC , review, approve and execute it .

- ➢ But if it is done at later stages we don't have time to write, review and approve , execute the TC as well as fix any bugs .



Requirement Traceability matrix Template :

- Refer to the excel sheet .

Test Execution :

The process of executing the application .

-Test Execution Process

- ➢ As soon as the build is ready the TE's start testing process
- ➢ In order to test , TE's take the TC's for their assigned module and start executing the steps of the TC on the build . this process of executing the steps of the build is known as Test execution process
- ➢ If the expected result and actual result is same , the particular step is said to be passed , if all the steps of the TC's are passed , the test case is said to be passed , if any of the steps of the TC is failed , the TC is said to be failed . if the expected result and actual result deviates , that particular step is said to be failed .

Note :

Sometimes , if most of the steps of the TC is passed with one r two minor steps as failed in such case the TC is said to be Partially passed TC

-Test Execution Report

It is a final consolidated report , prepared by the Leads after all the TC's are executed and all the required bugs are fixed except can't be fixed and deferred .

Generally , the execution report talks about total cases written for the module executed PASS / FAIL and their percentage

The execution report is a final report that defines the stability of the application .

Test execution report :

- Refer to the excel sheet .

Define the Testing Process / Procedure / Short End / STLC ?

➔ It is a step by step procedure to test the application and it is a part of SDLC.
➔ Initially , once we get the requirements the leads and managers do one round of requirement analysis and try to understand the complexity of the modules.
➔ Based on their understanding they prepare a document which acts as further plan for all the TE's.
➔ The test plan is mailed to all the TE's so that everyone can start their testing activities as per test plan.
➔ The first job of the TE's to prepare the necessary test documents during this process they follow a procedure
    1. First they try to understand the requirements and any doubts get clarified by the concerned person.
    2. Based on their understanding first they prepare a high level document known as Test Scenarios , which gives a brief idea about the flow of the application.
    3. After the test scenarios , we prepare a detailed document known as test cases which can be referred by any person who wants to test the application.

4. Our entire test execution depends on the test cases , hence it needs to be reviewed and approved to correct the mistakes.
5. Once the documents are ready all are kept in a centralised location known as Test case repository. So , anyone can across it if all is required.
6. During this process the developers are busy in writing the code and preparing the build.

- ➔ During the execution we may find some bugs / defects that need to be logged and tracked using some bug tracking tools.
- ➔ The Defect tracking continues until all the bugs are fixed and closed.
- ➔ Finally , we prepare a test execution report which talks about the stability of the application and then , the product is handed over to the customer along with necessary deliverables.
- ➔ During this entire process , we would have encountered some problems. These problems are discussed in a meeting known as Retrospect meeting so that we can avoid such issues in the upcoming releases.

RA Meeting :

RA is a requirement acceptance meeting. It includes that BDA conducts this meeting to check whether the requirements given by the client is doable or not and also check the complexion in the requirement in case of complexity.

Difference between SDLC and STLC :

SDLC	STLC
Software development life cycle	Software Testing life cycle
SDLC is a step by step procedure to develop the application.	STLC is a step by step procedure to test the application.
SDLC involves different teams to develop applications.	Only testing tea is involved.
SDLC is a complete life cycle.	STLC is a part of SDLC.
Phases	Phases

TEST PLAN

1. Test plan is a dynamic document prepared at the initial stage as soon as we get the requirement.
2. This document derives the entire testing activities.
3. It is prepared by the leads and managers.
4. It consists of various attributes such as

   Objective : it talks about the purposes or aim of preparing the Test Plan . This test plan is written to check the functionality of whatever release No.

   Note : we prepare a test plan for every release , plan changes as the release changes or cooriding to the need.

   Scope : it talked about the features which need / needn't to be tested in the current release of the application.

   Release 2 -------- Draft , Trash , Help .

   Release 1 ------- Login , Home , Compose , Inbox , Logout .

   Inscope : draft , trash , help , compose , inbox.

   Outscope : login , home , logout .

   Approach : It talks about the sequence in which the application needs to be tested /// it talks about the flow of the aplicacion .

   Login -- home -- compose -- SI -- logout .

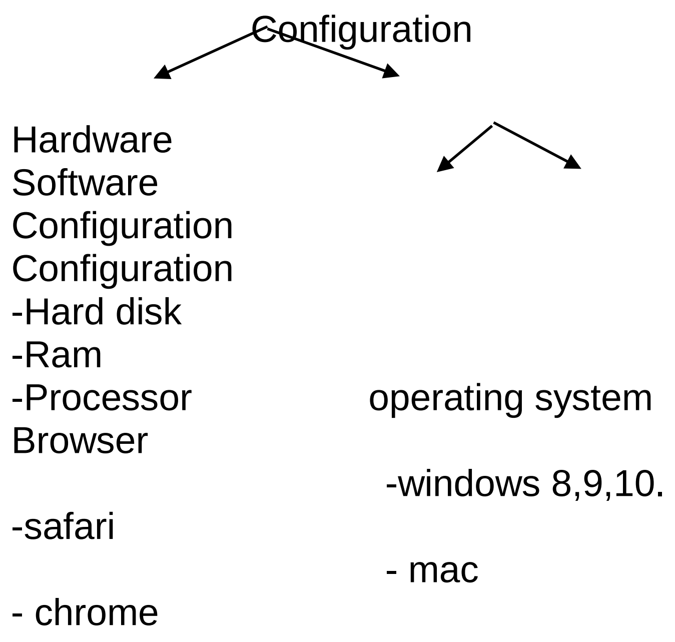
   Login -- Inbox -- logout .

   Test Methodology :it talks about the various types of testing that need to be performed in the current release of the application.

   Compulsory : smoke , FT , IT , ST , UAT

   On Customers demand : PT, CT, UT, Ad Hoc.

   Test Environment : it talks about all the necessary configuration which the application needs to be testing.



Templates : It is a predefined format used by every test engineer to prepare the test doc.

Test case template, test case review template , bug report , test execution report , RTM , are different types of templates.

Roles and Responsibilities :

Test Manager :

- Prepare or review the test plan.
- Conducting meetings with the customer.
- Conducting monthly status meetings with the team.
- Conducting meetings with the development team.
- Review and annual performance of the employees and also set annual goals.

Test Lead :

- Participate or prepare the test plan.
- Conduct daily stats meetings with the team.
- Assign the task to the team members.
- Track the day to day activities and help the team to overcome the challenges.
- Conduct meetings with the development team.
- Review the critical cases.

Test Engineer :

- Understand the requirement.

- Review the test documents.
- Review the documents , prepared by the other TE's.
- Perform test execution process.
- Log and track the bugs.
- Participate in various types of meetings.

<u>Effort Estimation</u> :it talks about the efforts which need to be applied by each team member to achieve the goal.

<u>Schedules</u> :it talks about the sequence in which the task need to be performed i.e, the duration as well the sequence

Smoke ----- FT ---- IT ----- ST ----- Regression ---- Auto ---- ……..

<u>Tools</u> : it talks about all the list of tools which are going to be used in the current release of the application , bug tracking tool , test management tool , functional , non functional testing tool .

<u>Entry and Exit criteria</u> :

Entry criteria :

- Build should be ready
- Test document should be ready
- Task should be assigned to the team
- Necessary resources must be ready

  TE // Laptop // Internet // Configuration // Server

Exit criteria :

- All test case must be executed
- Most of the test cases must be passed
- There should be no blocker bug , critical or major , few minor bug may exist
- All necessary deliverable must be ready

<u>Test Deliverables</u> : List of documents which needs to be handed over to the customers along with the application . This includes - test cases , test scripts , test scenarios , RTM, test execution report and release.

Release notes:

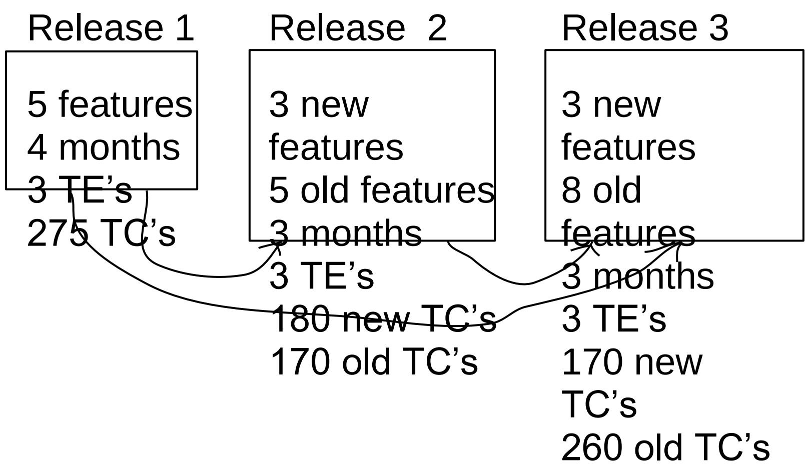
- User manual
- Installation guide
- Platform on which the application is tested

- List of existing bugs in the current release
- List of fixed bugs from the previous release

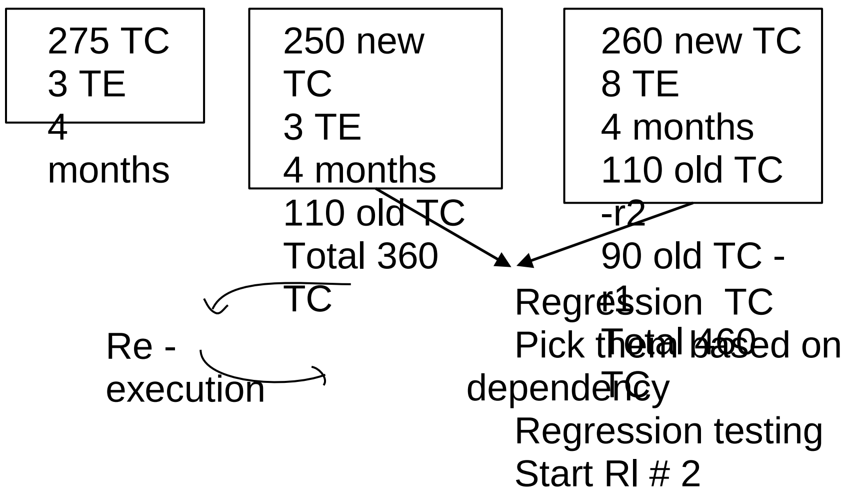
Assumption : There are the problems or issues which may be faced during the process.

Risk : This is the effect of assuming a problem of the testing.

Migration : Back-up plan prepared by test lead or test manager to overcome the assumed risk.



5 features	5 old features	3 old features
Rel # 1	4 new features	4 old features
	Rel # 2	7 new features
		Rel # 3 …………… Rel # n.



Code changes (addition , deletion , modification)

Addition : when we add new features we are using this.

Deletion : when we want to delete the code.

Modification : when we have the bugs in code we are modifying the code.

Challenges of Regression Testing :

1. Identify impact area
   - By doing impact analysis.
2. Time consuming
   - TC increases release by release
   - Less resources
3. No accuray
   - Repetitive / Monotonous task
   - Hectic

Regression Test Case :

Whenever the customer asks for change in the application and so the developer does the code changes (code changes may be addition, deletion or modification) which may affect the other features , generally pick the dependent features (i.e, dependent TC's) and do re- execution of the old TC's. These old TC's are known as regression TC's and the process of re-execution is known as regression testing.

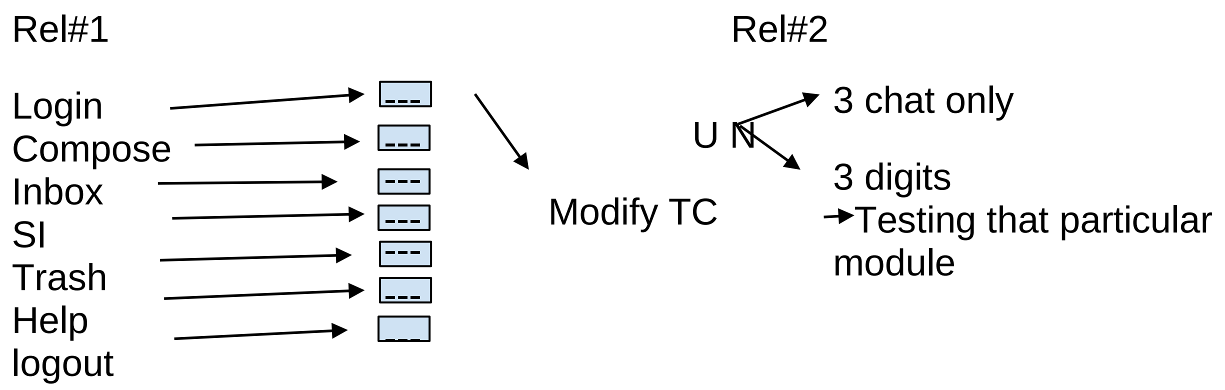
Types of Regression Testing :

1. Unit regression testing
2. Regional regression testing
3. Full regression testing

Unit Regression Testing :

Check only the changed feature + no impact area is known as Unit Regression Testing.

One feature which is Independent



- ➔ Due to any minor changes in the next release, are generally modify the old TC and save it in the new repository.
- ➔ The old copy of the old TC is deleted from the old repository.
- ➔ Regression testing is known as unit regression testing because, only one unit / module TC is modified and re-executed and it doesn't affect the other units / modules i.e, we dont touch any other unit / module TC's.

  Note :

  Whenever customers ask for any major changes we don't modify the old TC's instead of which we write a new test case.

AUTOMATION TESTING:

Checking the application with the help of tools is called automation testing.

Functional testing tools (Tools like selenium, QTP, silk test)

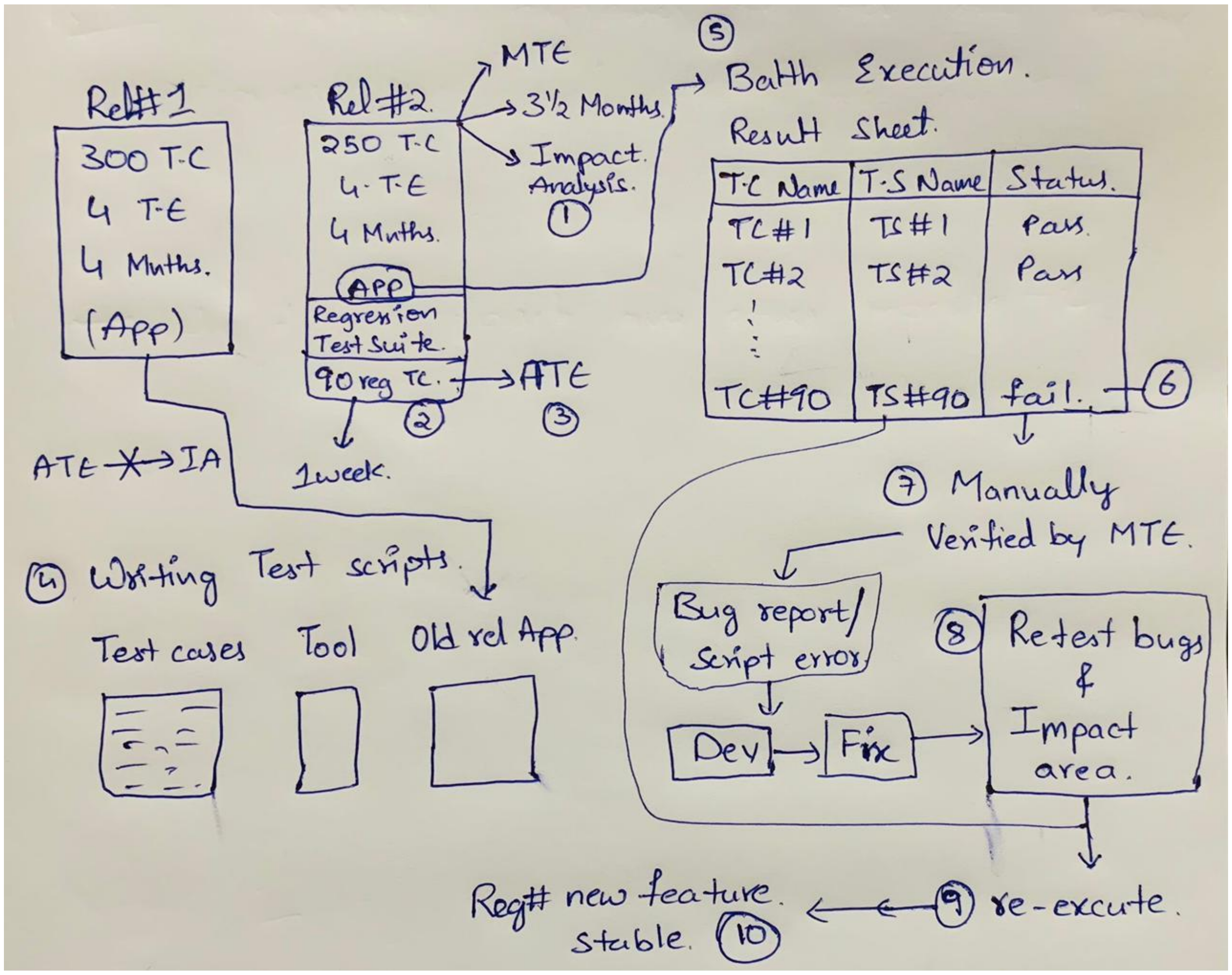
Non-functional testing tools (Tools like Load runner, Jmeter , neo load)

Bug tracking tools (Tools like bugzilla , zila , mantish)

Test management tools (Tools like ALM / QC , test manager)

- ➔ Regression testing by default means functional testing and we perform regression testing with the help of functional automation testing tools .
- ➔ Automation testing starts from 2nd release onwards because regression testing starts from 2nd release onwards .
- ➔ Whenever we have repetitive task we go for automation testing

Automation Testing Process :



1. Generally, we go for automation only when there are multiple releases or multiple regression cycles or repetitive tasks.
2. The process of testing the application by using tools is known as automation testing
3. From the release#2 , once we get the enhancement , first we do one round of impact analysis and prepare the regression test suite .
4. The regression test suite is handed over to the automation test engineer , since doing regression manually is always time consuming and not accuracy task
5. So, from the release #2, all teams work parallely like developer working on new features, manual TE working on new features and automation TE are busy in writing test scripts
6. During , writing of test scripts , they use the regression test cases , tool, and the previous release application , since new application is under development

7. By the time , manual team finishes working on the new features the automation team should complete writing the test scripts .
8. Once the new features are stable and scripts are ready the automation team starts executing the written scripts for regression TC's on the new application (old features)
9. Once we get the automation results and if any of the scripts gives the result as fail , generally manual team verifies it manually
10. The developer fixes the bug and we test the bug   with impact area (MTE) and automation team re execute the test scripts
11. This Process continues until the regression feature as well as the new features are stable.

Roles of Manual test Engineers in Automation Process:
1. Identify the impact area
2. Create impact area report to help lead to prepare regression test suite
3. Help to avoid the major risk
4. Re-execute the cases which are failed in the batch execution
5. Log the bugs and track the status
6. Check the impact area due t the bug fixes

The Advantage of Automation Testing are :
1. Accuracy exists : Since tools do not have any brain , so they don't get tired or bored of being repetitive and they will not skip any values or scenarios in the test case .
2. During regression testing with automation is always time saving process , because
    - Execution happens very faster than manual
    - We can always go for batch execution .i.e, all the scripts can be executed simultaneously
3. Once the test scripts are ready , we can re-use or re-execute them whenever the developer does the code changes --in the next release .

Challenges Of Automatic Testing :
1. New test case cannot be automated
2. Sometime writing the script for some scenarios may consume more time with respect to do testing manually
3. Some test cases cannot be automated where human involvement is required such as OTP , dynamic Captcha etc..

Questions in Interview :
1. What is Automation testing
2. Why (advantages)
3. When
4. Process
5. Challenges --- which all test cases cannot be automated

6. Roles of MTE in the automation process.

Difference between Regression Testing and Re- Testing :

REGRESSION TESTING	RE- TESTING

Smoke testing :

Approach 1:

As soon as the build is ready and installed in the test server, the TE's start checking each and every feature one by one with various combination of inputs (rigorously) (Functional Testing)

Build 1 -- Login - Compose - Inbox - SI - Logout

TE will start testing login module --- 2 days , on 3rd day they start testing the Compose module

TE click on compose and they get a blank page -- report it to the developers --- now the developer will take 2 days of time to fix it .

TE will sit idle

Developer will be under pressure

Test cycle will increase

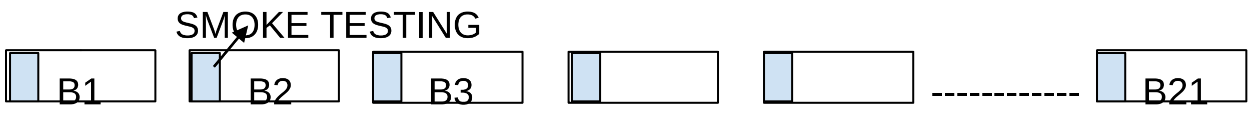
Approach 2 : -- Smoke testing /

As soon as the build is installed in the test server , the TE's first start checking all the features with one round of positive inputs
Build 1 -- Login - Compose - Inbox - SI - Logout

TE will start checking Login module --2hrs(create an a/c and login with valid credentials)

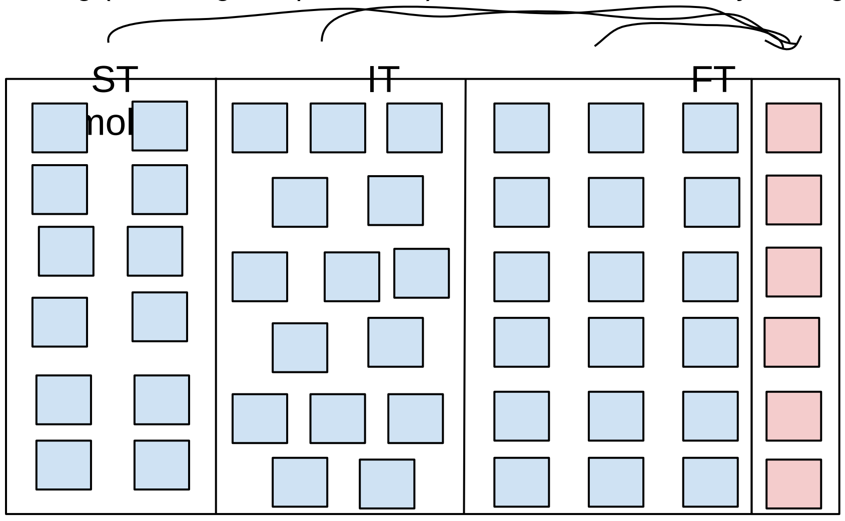
Click on compose , a blank page will display so within 2hrs we will encounter the blocker bug in the application , report it to the developer and TE will continue the testing process with various combinations of inputs for the login module and meanwhile the developers will fix the bug .

1. TE will not be idle
2. Developer will not be under pressure
3. Test Cycle will not increase
4. Blocker bugs are encountered at the early stage



Checking the basic and critical features of the application before moving into deep rigorous testing is known as Smoke testing.
We go for smoke testing as soon as the build is installed to ensure the build is testable or not hence it is also known as build verification testing .
Smoke testing is done for every build as soon as it is installed followed by other rigorous testing.
With the help of Smoke testing we can encounter the blocker bug at very early stage and Hand it over to the developer meanwhile the TE will not sit idle and start testing other independent testable modules.

Note :
Smoke testing / Sanity testing / Dry Run testing / Skim testing / Build Verification Testing .
Smoke testing is not a compulsory type of testing, it is a good approach to identify the blocker bug at an early stage.
When we go from build 4 to build 5 -- perform impact area testing --build level regression testing (retesting + impact area) ---this also called sanity testing .



Test Case Repository

Difference between Smoke Testing and Sanity Testing .

SMOKE TESTING	SANITY TESTING
We go for smoke testing as soon as the build is installed in the test server.	We go for sanity testing as soon as the bug is fixed.
To check if the build is testable or not.	To check the impact area due to bug fixes.
Smoke testing is shallow and wide.	Sanity testing is deep and narrow.
Smoke testing can be documented and scripted since we can identify the basic and critical features of the application.	Sanity cannot be documented and scripted since we cannot assume or predict the bug.

Difference between the Service Based Company and Product Based Company:

SERVICE BASED COMPANY	PRODUCT BASED COMPANY
The service based company who develops the product by collecting business requirements.	The product based company who develops the product and sells it in the market.
The applications developed are particularly developed for particular customers.	In a product based company the product is developed and sold to multiple similar customers.
In a service based company the customer is the owner of the product.	Whereas in the product based company it is the owner of the product.
In service based companies along with the application , related docs such as test cases, source code is given to the customer.	In product based companies only the application along with the user manual and installation guide is handed over to the customer.

Service based Company ----
Say TCS ---- 2 lakhs----
250 projects ---- 250 customers---
Requirements -----
Recession ---- No project --- No Money---
What should TCS do ?
If fire the employees --- 40000 -- 2 months salary should pay--
6 months market will increase ---- clients
---once again to hire resources ---

Product Analyst ----- Market Analysis ---
Bring the requirement from the market by studying market needs----

Business Bodies :-
Eg : Hospital ----PA ---
Study the hospital ----- say in Manipal ---
Challenges , the hospital is facing ---
--- Doctor Management application ---
--Nurse Management application is missing --

PA will collect the requirements by the market analysis ---\
No customer is giving the requirement ----
Based on the requirements given by the PA the development team will develop and testing eam will test the application ---
Requirement may not be clear ---
---- Exploratory testing ---

Exploratory Testing :
In exploratory testing we explore the similar existing application to test the application.
In this type of testing theTE prepares the test document and uses it to perform the test execution process.

-Disadvantage
Misunderstood the requirements as a result sometimes features may be treated as bugs and sometimes bugs may be treated as features.

Adhoc testing :
After performing all the types of testing(smoke/ FT/ IT / ST /non functional) and application is stable then ,We will test some random scenarios..

1. I compose a mail with attachment , by the time attachment is getting attached, click on send button.
2. Transferring the amount to someone, once a transaction is going on close the browser.
3. Whatsapp messages are getting sent and power off the phone.

Checking the application randomly without following any sequence or rule is known as Adhoc testing. Once all the functional as well as non functional (if required) testing is over and still we have some time in such cases we TE ,check for the random scenarios. Adhoc testing helps the customer to know the consequences of such a random scenario.

Note :
  Adhoc testing is not a compulsory type of testing , since in this type of testing we check only the random scenarios of a stable application. We go for it only if time permits.

We wantedly test the application randomly
In exp we want to test the t

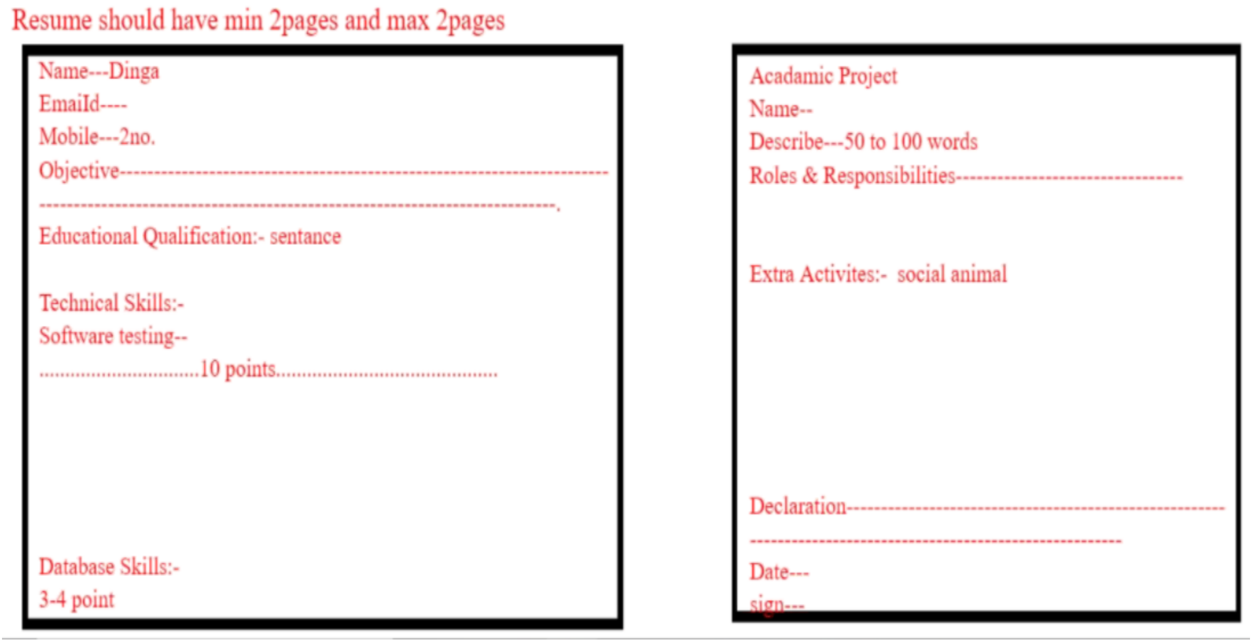
Alpha testing and Beta testing :
Alpha and beta testing are the terminologies of product based company.

Alpha testing is similar to all the types of testing done by the TE’s inside the company in service based company.
Beta testing is similar to that of UAT done by the clients in a service based company .

ROLES AND RESPONSIBILITIES OF MANUAL TEST ENGINEER

1. Go Through the requirements.
2. Understand the flow of the application.
3. Participate in writing the test plan.
4. Writing the test documents (test cases).
5. Review the test cases (peer review).
6. Prepare RTM on your assigned modules.
7. Cross verify the test cases in the case repository.
8. Start test execution process.
9. Identify the bugs and log it to the concerned developer.
10. Knowledge of handling the bug tracking tools.
11. Track the status of the bugs until all the bugs are closed.
12. Participate in a meeting with the development team (if required).
13. Prepare test execution report on assigned modules.
14. Test the application across multiple OS and browsers.
15. Participate in various types of meeting conducted by the team
16. Participate in preparing the test deliverables.
17. Perform impact analysis process.
18. Identify the major risk based on your product knowledge.
19. Discuss the major risk with the team and solution for it.
20. Help the lead to prepare the regression test suite.
21. Perform sanity across the builds (retesting of bug + impact area).

Resume for freshers:



Resume should have min 2pages and max 2pages

Name---Dinga
EmailId----
Mobile---2no.
Objective--
---.
Educational Qualification:- sentance

Technical Skills:-
Software testing--
................................10 points..

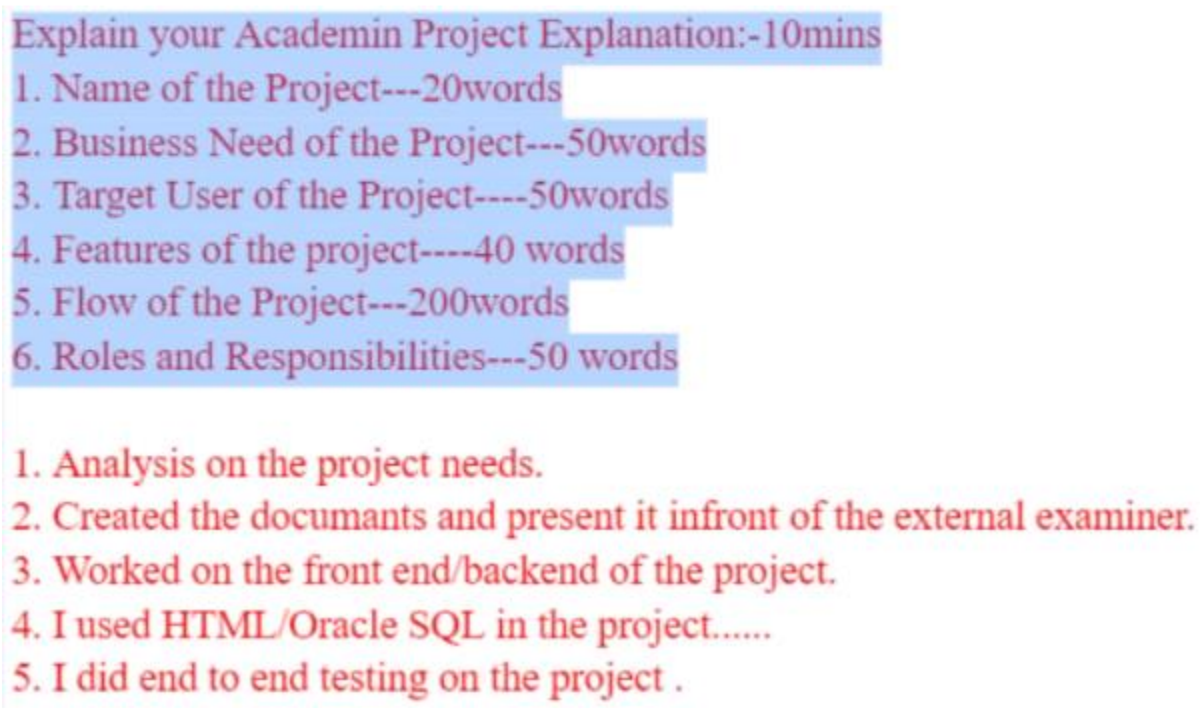
Database Skills:-
3-4 point

Acadamic Project
Name--
Describe---50 to 100 words
Roles & Responsibilities---------------------------------

Extra Activites:- social animal

Declaration--

Date---
sign---



Explain your Academin Project Explanation:-10mins

1. Name of the Project---20words
2. Business Need of the Project---50words
3. Target User of the Project----50words
4. Features of the project----40 words
5. Flow of the Project---200words
6. Roles and Responsibilities---50 words

1. Analysis on the project needs.
2. Created the documants and present it infront of the external examiner.
3. Worked on the front end/backend of the project.
4. I used HTML/Oracle SQL in the project......
5. I did end to end testing on the project .