

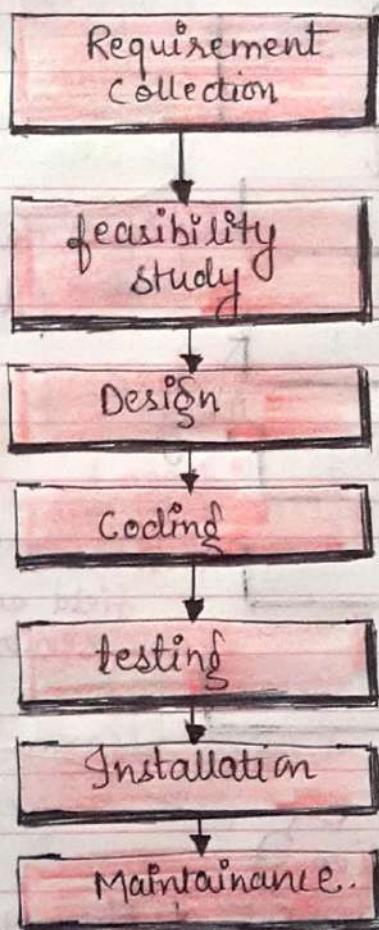
# **Manual testing**

**@Harish Kumar**

# SDLC

software development life cycle

- \* It is the standard procedure to develop the software.
- \* It has different stages:-

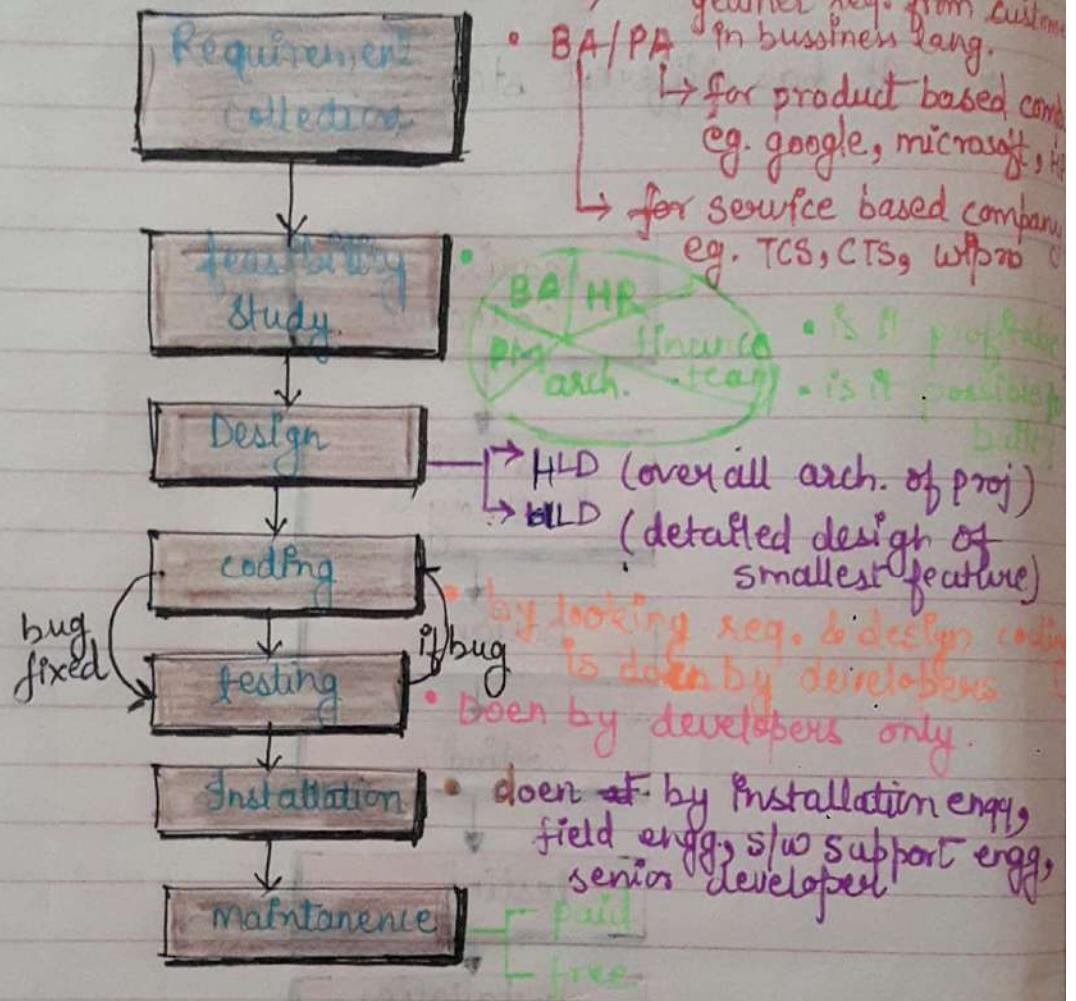


Harish Kumar

- \* Different models of SDLC are:-
  - 1). Waterfall model
  - 2). Spiral model
  - 3). V model
  - 4). Prototype model
  - 5). Derived model
  - 6). Hybrid model
  - 7) Agile Model (used in current working project)

# Waterfall Model

- \* It is also called traditional model or basic model.
- \* Stages of this model are :-



## Requirement collection :-

In this phase requirement of the customer will be collected by either BA (business analyst) or PA (product analyst) in some business language. And then he will convert it into software language and explain it to developers, project manager, architect etc. Here BA acts like a bridge between customer & software company. BA will be hired, only if they have many years of experience in the specific field & they get trained & then sent to the client place.

who can become BA?

- 1). Domain expert (The person who worked on same domain for 10-15 y & has got very good knowledge)
- 2). Senior developer (8-10 y of exp. In the same sort of proj. & has got very good project knowledge)
- 3). Senior test engg. (8-10 y of exp. In the same sort of proj. & has got very good project knowledge).

### Feasibility study:

following group of people it's & decide.

BA: who converts business language to software language

Architect: who checks whether the project is technically feasible?

HR team: HR team decides whether there is a resource needed & how many needed?

Finance team: In the finance team they used to check the profit% or loss if they gets the needed profit%. they approve.

Project manager: All the people who send the report to the project manager about their analysis. He is the final decision authority of the project.

Once software requirement collection is completed then only we go for feasibility study. This is done by a team which consists of pm, BA, HR, architect, finance team. This is the stage where company will decide whether

to take up the project or not. If we take up the project company checks do we have sufficient resource, sufficient technology & sufficient lab setups.

This is the stage where company will get to know if they take up the project, do they get profit or not.

## Design:

Designing of the project is done by either Architect, senior developer, as well as content design & web designer also.

Design is of 2 types

- HLD (high level design)  
where we design the overall architecture of the project.
- LLD (low level design)  
where detailed design of the smallest feature is done.

## Coding

By looking on to requirement & design of the project line-by-line code is written for each module by set of group of developers (senior developers, junior developers & fresher developers)

Senior developer: he will handle critical section (daily used frequently)

Junior developer: he will handle major section (e.g. edit, forgot password)

Fresher: he will handle minor (e.g. about, help, profile)

# Harish kumar-sk

## Testing

Here the application / project is tested & if there is any bug it is reported & sent back to the developer to fix it. Once fixed it's retested until the SW gets stable. It is again done by the set of testers (Senior tester, junior tester, fresher testers).

## Installation

In one time installation, senior developer will do the installation.

In case of many installation (e.g. health app.) - field engg / Installation engg. will do the installation.

Once SW testing is completed & if SW is ready we will install the SW in customer place so that the customer can use the SW & run the business.

## Maintainence

Once the SW is installed in customer place customer will use the SW & run the business. While using the SW, customer can face any problem, company will fix the defect for freely (for limited period e.g. 1-6 months) & will cost some amount after that fixed period or for major change depending on the agreement the SW company & the customer signed before this phase is called maintenance phase.

- CRS (Customer requirement specification)
- BS (Business specification)
- BRB (Business requirement specification)
- BRD (Business requirement document)
- SRS (Software requirement specification)
- FS (Functional specification)

### Advantages :

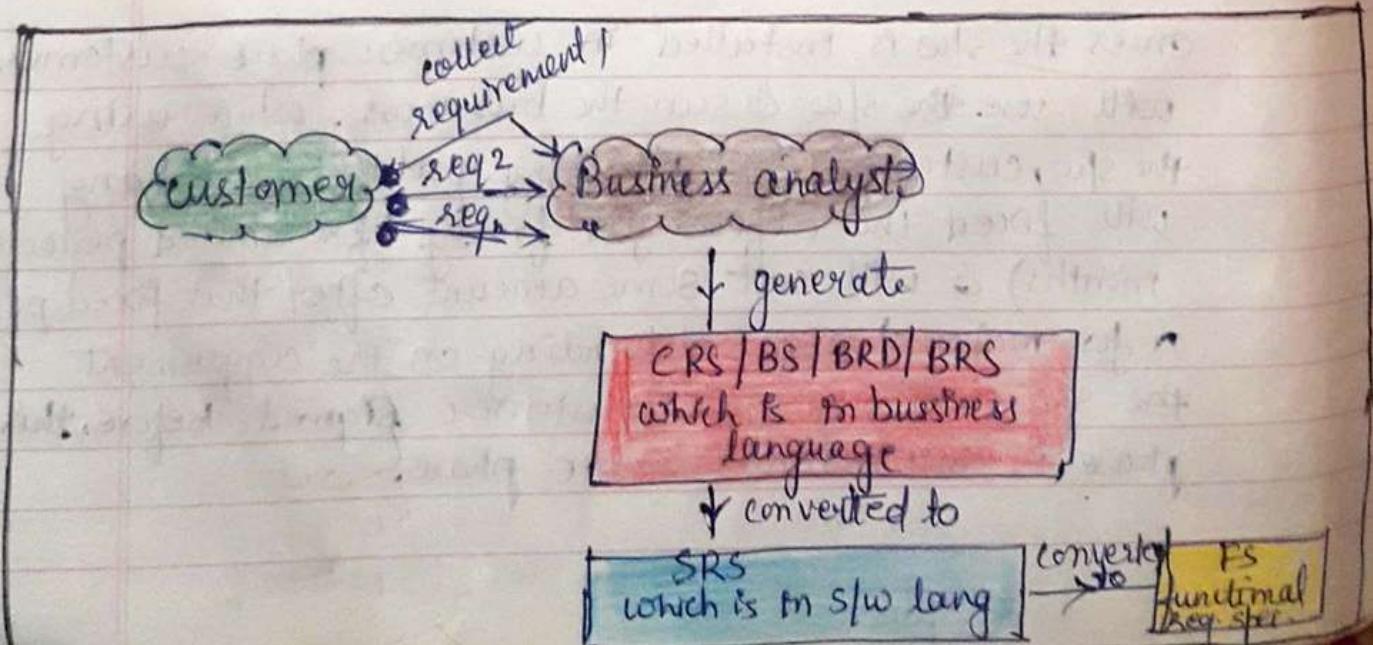
- ①. Simple model.
- ②. Initial investment is less
- ③. As requirements are freezed so we can expect good quality product at the end.
- ④. very fast model.

### Disadvantages :

- ①. Only developers are evolved for testing.
- ②. Testing is done only after coding is completed i.e. there the requirement & design phase is not tested.
- ③. No back tracking can be done.

### Applications:

- for simple project
- for short term project
- for small applications
- for projects where requirements are freezed at the beginning only.



# Spiral Model

- \* It is a systematic approach for developing the software, it has 4 phases i.e. Requirement collection, Design, coding & testing.

Design

- D<sub>D</sub>
- D changes in B

minor changes in B

Requirement collection

R<sub>D</sub>

- changes in B
- 3 defects in C

- C<sub>D</sub>
- C changes in B
- fix 3 defects in C

C changes in B & C<sub>B</sub>

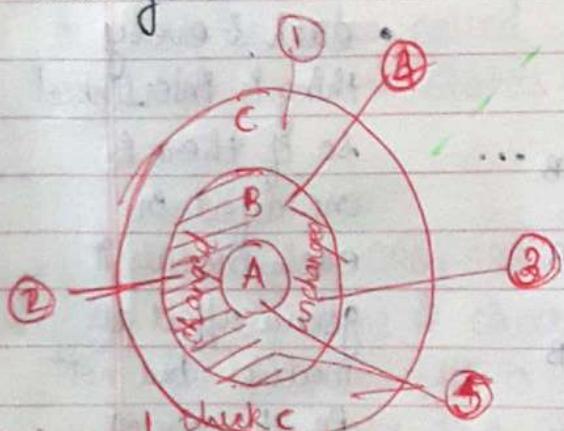
R<sub>B</sub>  
R<sub>A</sub>

LDA

CA

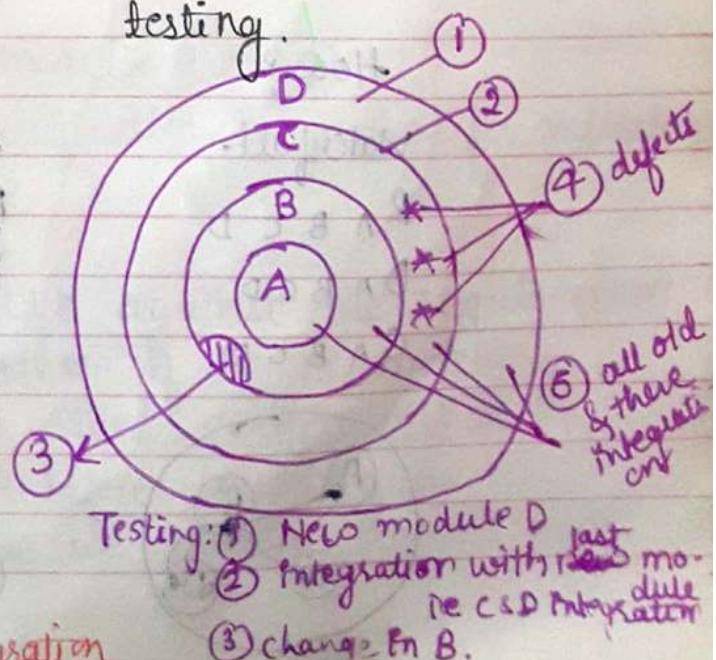
CB

Coding



- Testing:
- 1. changed B part
  - 2. unchanged B part
  - 3. integration of B & C
  - 4. all old modules & their integration

testing.



- Testing:
- 1. New module D
  - 2. Integration with last module C & D integration
  - 3. changes in B.

- \* Whenever there is dependency in between the modules of the project we go with spiral model.
- \* In this model we can do changes once after the module is completed.
- \* In order to change the design the complete cycle is used for the changes.
- \* It can handle both minor as well as major changes, in order to do minor changes in the older module, the customer will give the new module & also the minor changes in the older module.

### Advantages:

- 1). Requirement changes can be handled (whether they are minor or major changes).
- 2). It is a controlled model.

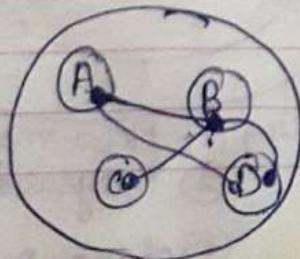
How?

Waterfall -

R A B C D

D A B C D

C A B C D



Spiral

R A R B ...

↓ ↓

D A D B ...

↓ ↓

C A C B ...

↓ ↓

T A T B ...

↓ ↓

A B

• Each & every thing is interlinked so if there is one defect in each module it gonna affect all modules but not in the case of spiral model.

- 3). Customer gets an opportunity to see the s/w in every stage.
- 4). Customer gets an opportunity to ask for the changes in every cycles.

Drawback - 1, 2 of waterfall.

### Applications

- 1). Whenever there is dependency in the modules.
- 2). Whenever the customer gives the requirement in stages.

## V-Model

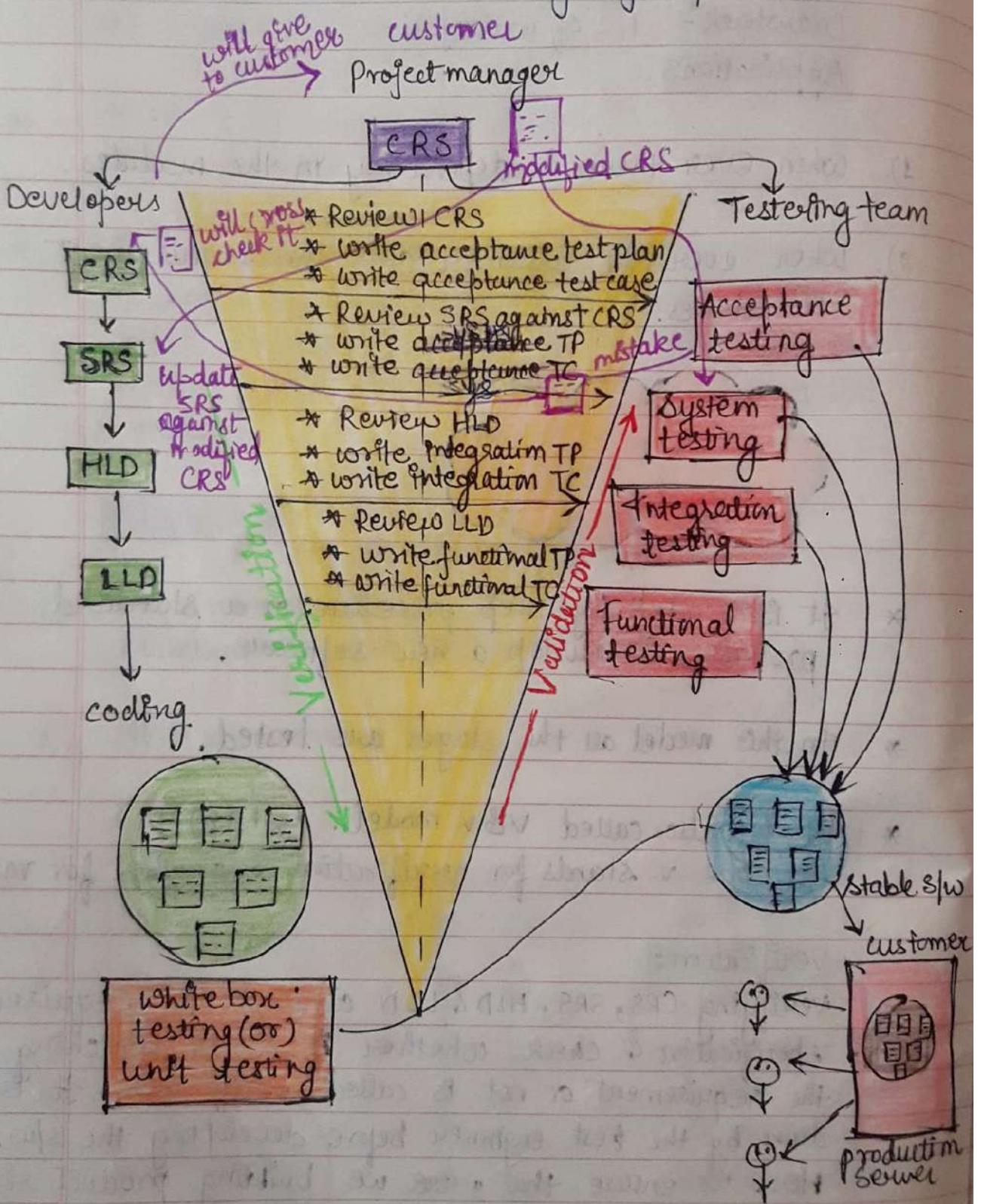
- \* It is a step-by-step procedure or a standard procedure to develop a new software.
- \* In this model all the stages are tested.
- \* It is also called V&V model. (why?)  
here one V stands for verification & another for validation

### Verification:

Verifying CRS, SRS, HLD & LLD against the requirement specification & check whether it is matching the requirement or not is called verification. It is done by the test engineer before developing the s/w. Here we ensure that, are we building product right?

## Validation:

Testing the functionality of an application by executing the test cases is called validation. It is done by test engineer once S/W is developed. Here we ensure that are we building right product.



\* name v-model is given becoz of the V shape, we get we for whole all ~~it~~ get.

Firstly the CRS is prepared by BA and then the project manager provides this CRS to both the teams ie. Development team & testing team.

Now Developer will start converting this CRS to SRS and test engg. will simultaneously Review the CRS and will write acceptance Test plan & ~~the~~ acceptance test case. & thus they while reviewing the CRS they may. get defects like.

- 1). missing requirement
- 2). conflict requirement
- 3). wrong requirement

### Missing requirement

Any requirement that is mentioned but not explained is called missing requirement.

e.g.



Page: OD will be used (overdraft feature)  
- but not explained in whole CRS

### Conflict requirement:

e.g.:

Loans.

5<sup>th</sup> page: Loan manager should not approve loan.

15<sup>th</sup> page: <10k - Loan manager should approve.

>10k < 50k - Branch manager --

>50k - Regional manager: --

e.g.:

Health care application.

5<sup>th</sup> page: followup fee < consultant fee.

15<sup>th</sup> page: followup fee = consult fee.

## User requirements

eg.

Type of loan

Home loan

personal loan

car loan

Amount

- Fixed interest
- floating interest

Submit

Cancel

copy

Paste

In SRS

fixed interest

- when ever user apply the loan through fixed interest, same interest should be charged till he repays.

Floating interest

- when ever user apply the loan through floating interest same interest should be charged till he repays.

wrong req

Now we will send this mistake to development team, then they will cross check it & if its a mistake they will send back to customer. customer will update the CRS & send back. Now with updated CRS developer's will update the SRS & test engg's will again review the updated CRS. this cycle continues till the test engg's approve it all alright, here 1st phase is completed.

- Now, In second phase developers will be involved in converting SRS to HLD till then testing team will review the SRS against CRS & will write system test plan & system test case.

while req reviewing SRS against CRS the test engg. may find defects like:

# Harish kumar

- 1). wrong conversion
- 2). Missing conversion
- 3). Not converted.

e.g.

should be radio button not check box.

Type of loan	<input checked="" type="checkbox"/> HL <input type="checkbox"/> PL <input type="checkbox"/> VL
Amt	<input type="text"/>
<input type="checkbox"/> fixed Int <input type="checkbox"/> float Int	
<input type="button" value="Submit"/>	<input type="button" value="Cancel"/>

select mistakes.

If mistakes are there, PT will be send back to development team. then they will fix PT & send back to the testing team.

Now in third phase, the developers will converted HLD to LLD & testing team will review HLD as well as will write integration test plan & integration test case.

In forth phase, the developers will be involved in writing code for LLD and the testing team will review LLD & will write functional test plan & functional test case.

Now the developers will do white box testing on the code they have written & will send to the testing team. And testing team will apply all the written test cases & test plane on the given application & will perform functional testing, Integration testing, system testing & acceptance testing. If they find

any defect it will be reported back to test devl. opens simultaneously. Then the developers will fix it & send the build back to testing team to retesting & it will continue till the software application becomes stable.

Once the application becomes stable, it will be send & handed over to the customer. And will be installed on its production server from where all the end users can use it.

So, here verification is done before the s/w is developed & validation is done after the s/w is developed.

### Advantages:

- 1). Testing starts in the early stage of product development, this avoids downward flow of defects which reduces lots of rebug.
- 2). All the stages are tested.
- 3). Deliverables are simultaneous, means developers are delivering SRS same time testing team delivering acceptance TC & TP so on becoz of which work gets completed faster.
- 4). Total investment is less.

### Drawbacks:

- 1) Initial Investment is high.
- 2) Involves too much documentation work.

### Application:

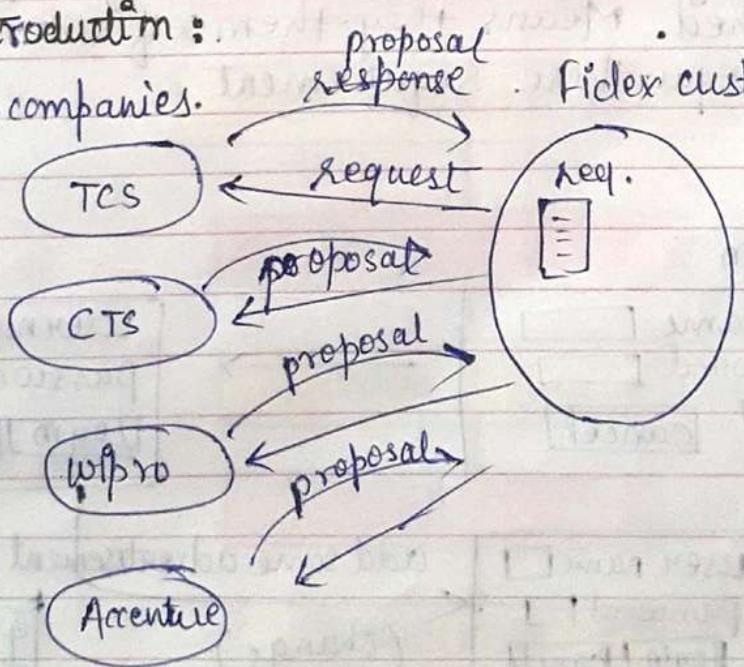
- 1). whenever the customer are expecting very high quality of product within a tight schedule.
- 2). whenever we are creating a complex projects
- 3). whenever we are developing long term projects.

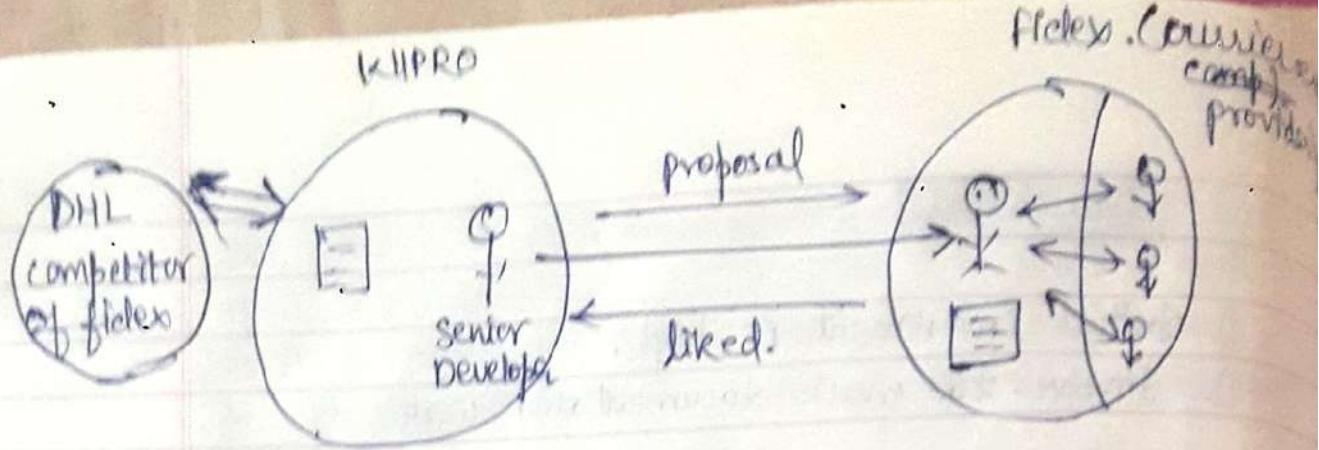
## Prototype Model

### Introduction:

S/w companies.

Fidex customer



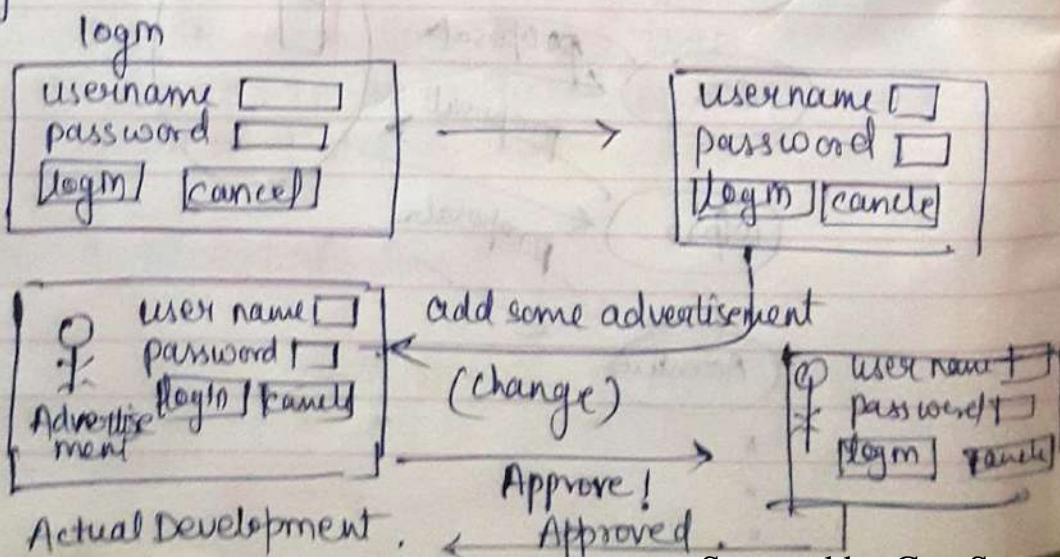


fidelx liked wipro's proposal (as wipro has thrown the basket that we have just made - the project for your competitor & we can develop something more better for you).

Now senior developer of wipro will go to fidelx & try collecting the requirement but some employ there say the & suppose he is collecting requirement for cancellation of service.

Now some of the employ says 50% of the amount should be refunded some says 25% of the amount should be refunded some says no charge should be taken if its not dispatched; some says 100% should be charge if its dispatched. Means they themself are confused with regard their requirement.

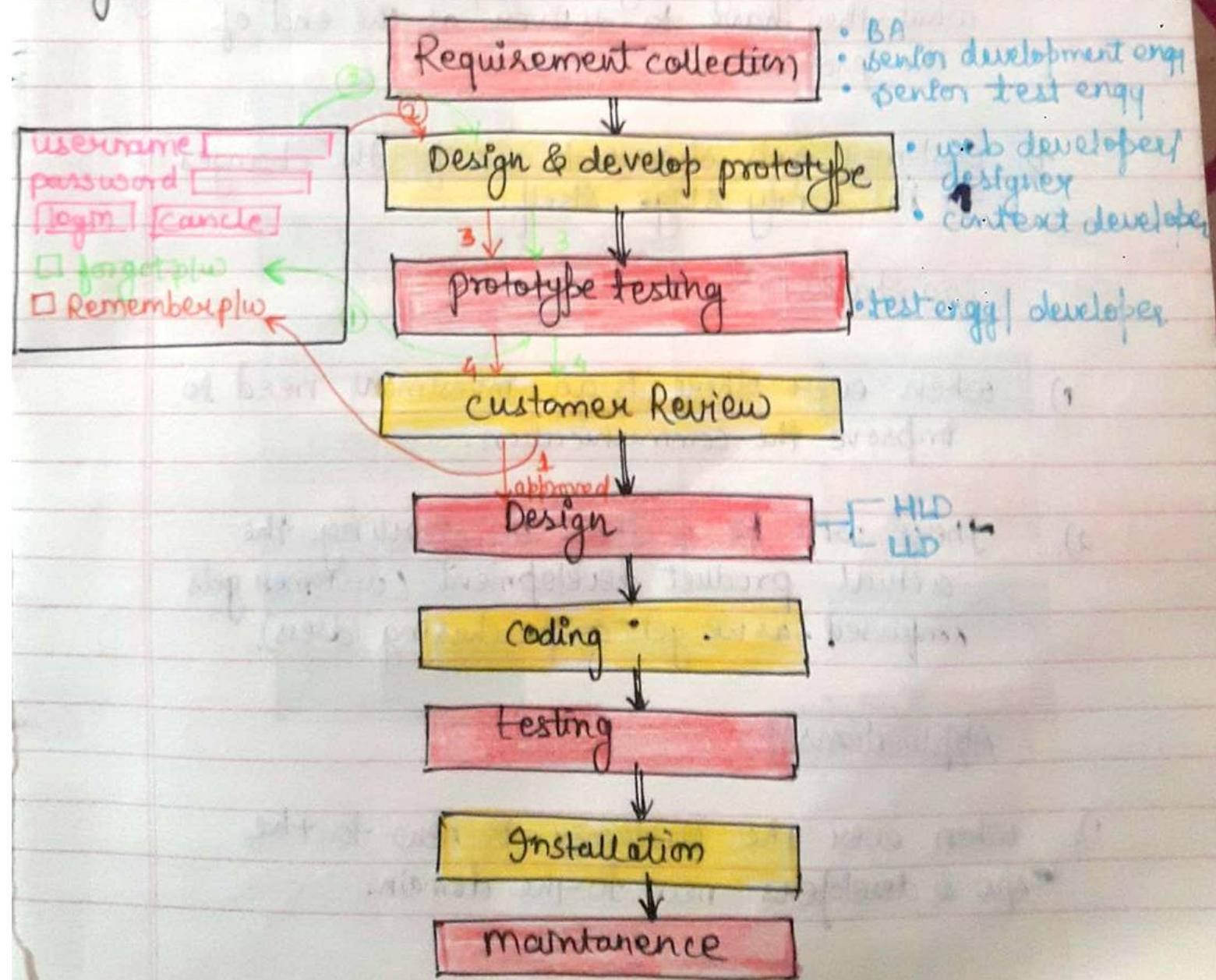
eg2.



prototype model is used where ever the customer has huge expectations but does not have clarity in the requirement.

thus first prototype (template) is send to the customer, if it gets approved by him then only the real develop design, coding & testing is done. And a customer can even deny later that what we developed is not what he wants. So its better to get approved by showing him the prototype of the project so make him clear, that how his project is going to look later.

Stages:



In prototype testing we check whether all the components are present according to requirement & look & feel of the prototype.

### Advantages:

- 1). There is a improved communication b/w customer & b/w company.
- 2). In the begining only customer is known what he gonna get at the end of the proj.
- 3). In the begining only developer will be knowing what they have to deliver at the end of the project.
- 4). Customer gets chance to ask for the changes in the early stage itself.

### Drawbacks:

- 1). When ever there is an investment need to improve the communication.
- 2). There will be a delay in starting the actual product development (customer gets confused - as we get on purchasing dies).

### Applications

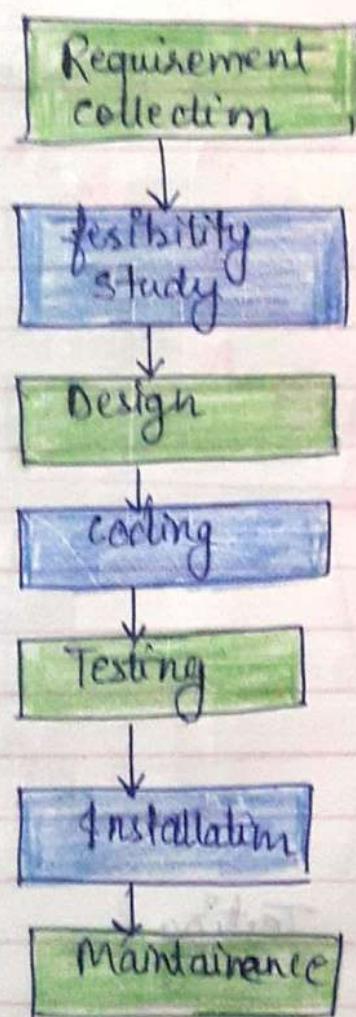
- 1). When ever the customer is new to the b/w & developers new to the domain.

- \*) whenever the customer is not clear about his own requirement.

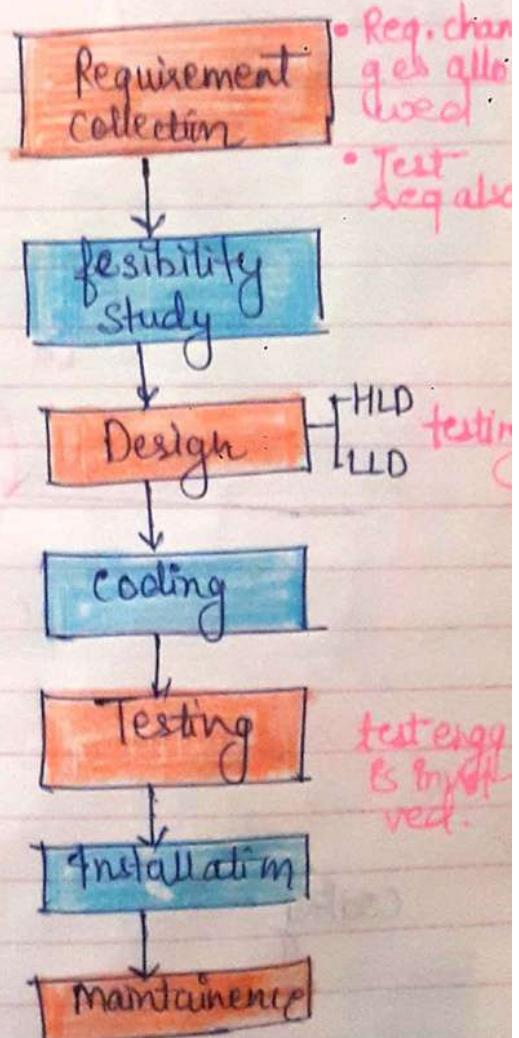
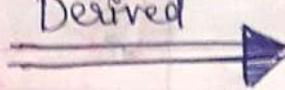
## Derived Model

- \* Here we take basic model & change it according to business & com. requirement & company standards, it is called derived model.

e.g.



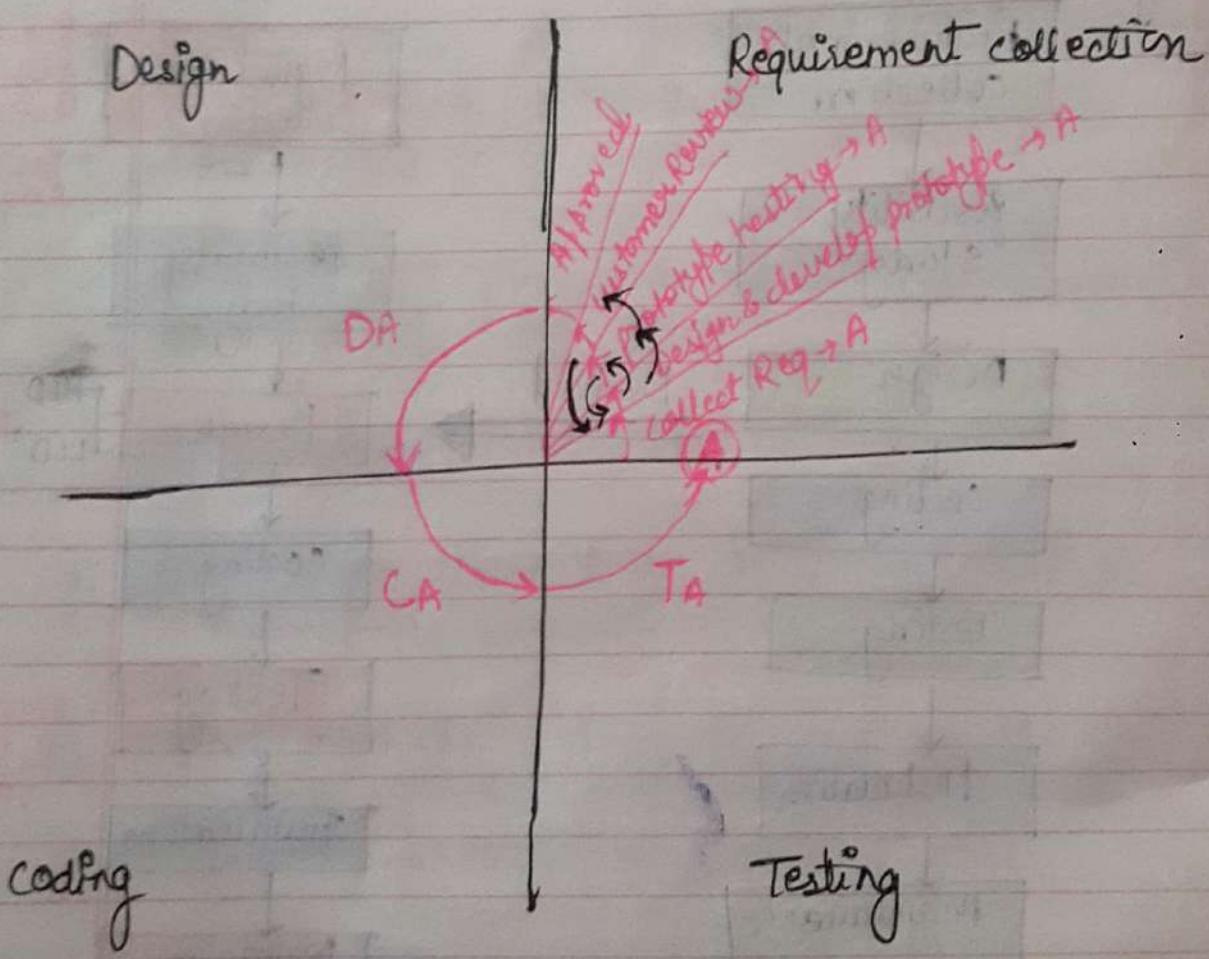
Derived



# Hybrid Model

- \* The process of combining or merging more than 1 model into a single model is called as hybrid model.
- \* Will get the advantage of both the models we have combined together.

Eg: spiral & prototype model

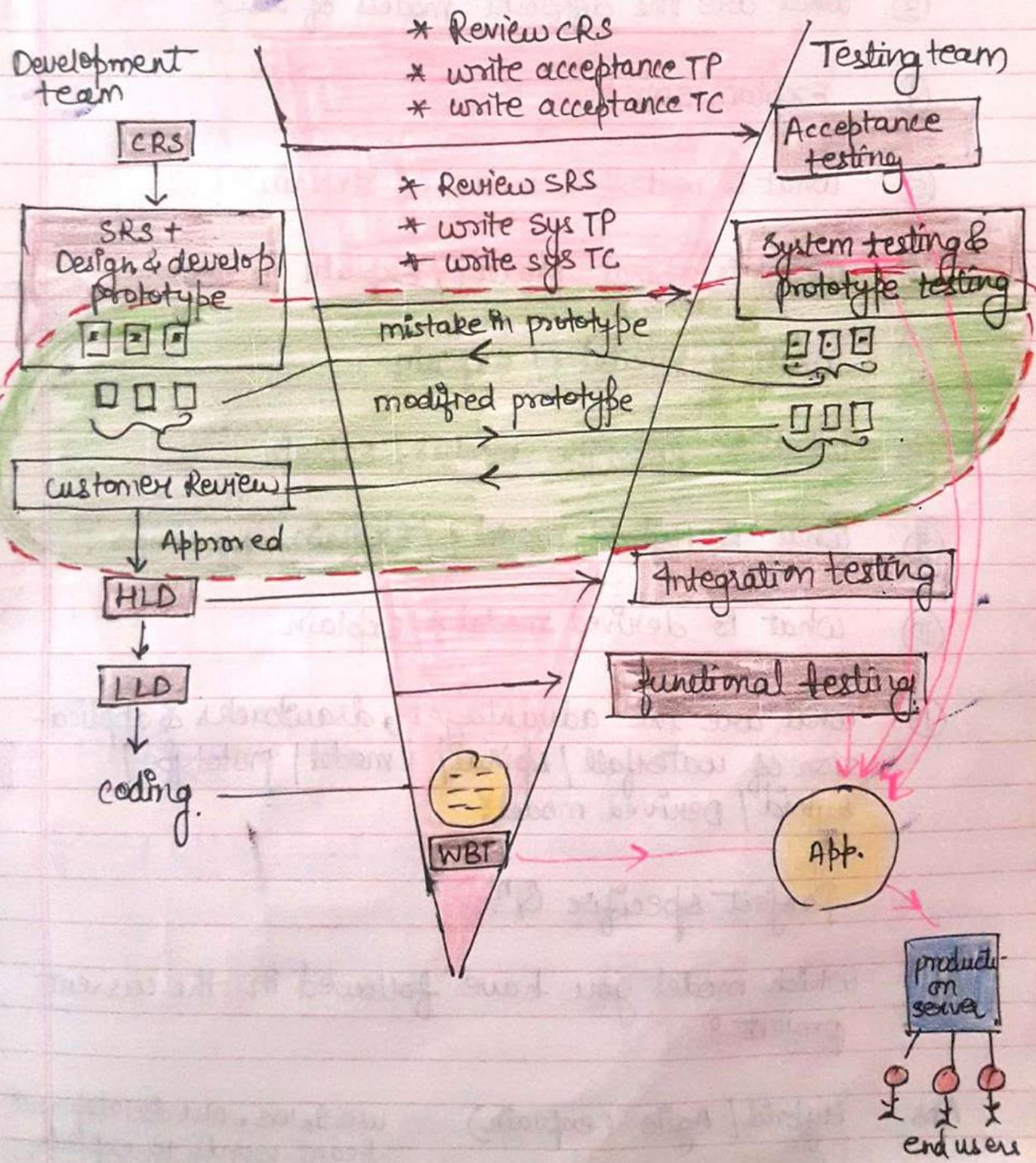


## Application:

When ever there is dependency between the models modules & when ever the customer

is not clear about his own requirement, when customer gives the requirement in stages we go for the combination of spiral & prototype model.

Eg: V & prototype Model:



## Generic Qn

- ①. What is SDLC?
- ②. What are the different stages of SDLC?
- ③. What are the different models of SDLC?
- ④. Explain SDLC?
- ⑤. What is waterfall model? / Explain
- ⑥. What is spiral model? / Explain
- ⑦. What is Vmodel? / Explain
- ⑧. What is prototype model? / Explain
- ⑨. What is Hybrid model? / Explain
- ⑩. What is derived model? / Explain
- ⑪. What are the advantages, drawbacks & application of waterfall / spiral / Vmodel / prototype / hybrid / Derived model?

## Project specific Qn

- ①. Which model you have followed in the current project?

Ans. Hybrid / Agile (explain)

use I, we, our development team words to explain the same & create the environment.

# Software Testing

It's an easy job but serious.

e.g.

qspider student

concept:

2 x (not attended)

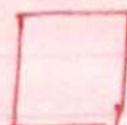
3 x

4

: x

got placed  
in NOKIA

Tested N8 phone



↓ user



call receiving button not working

switched off the phone

getting call from Dingi

got breakup

reputation of N8 goes down

e.g. Health care app.

→ appointment → consultation

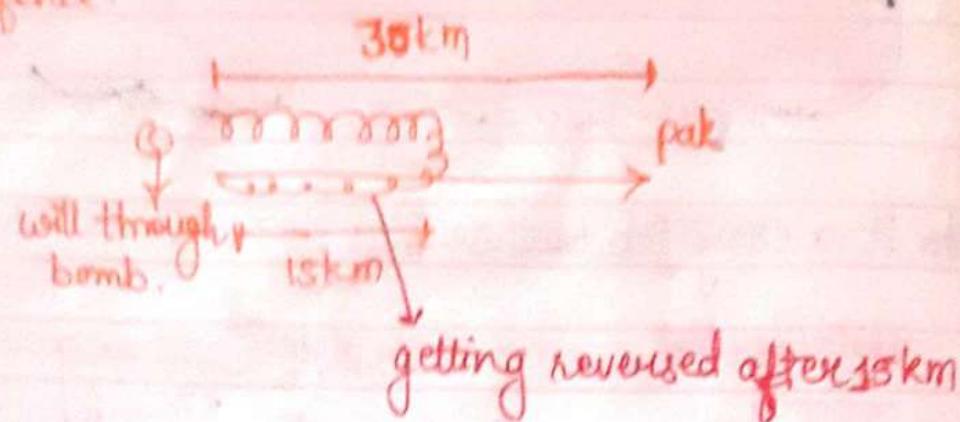
Dolo 650 mg 1-1-1 6 day

[submit] [cancel]

↓ pharmacy

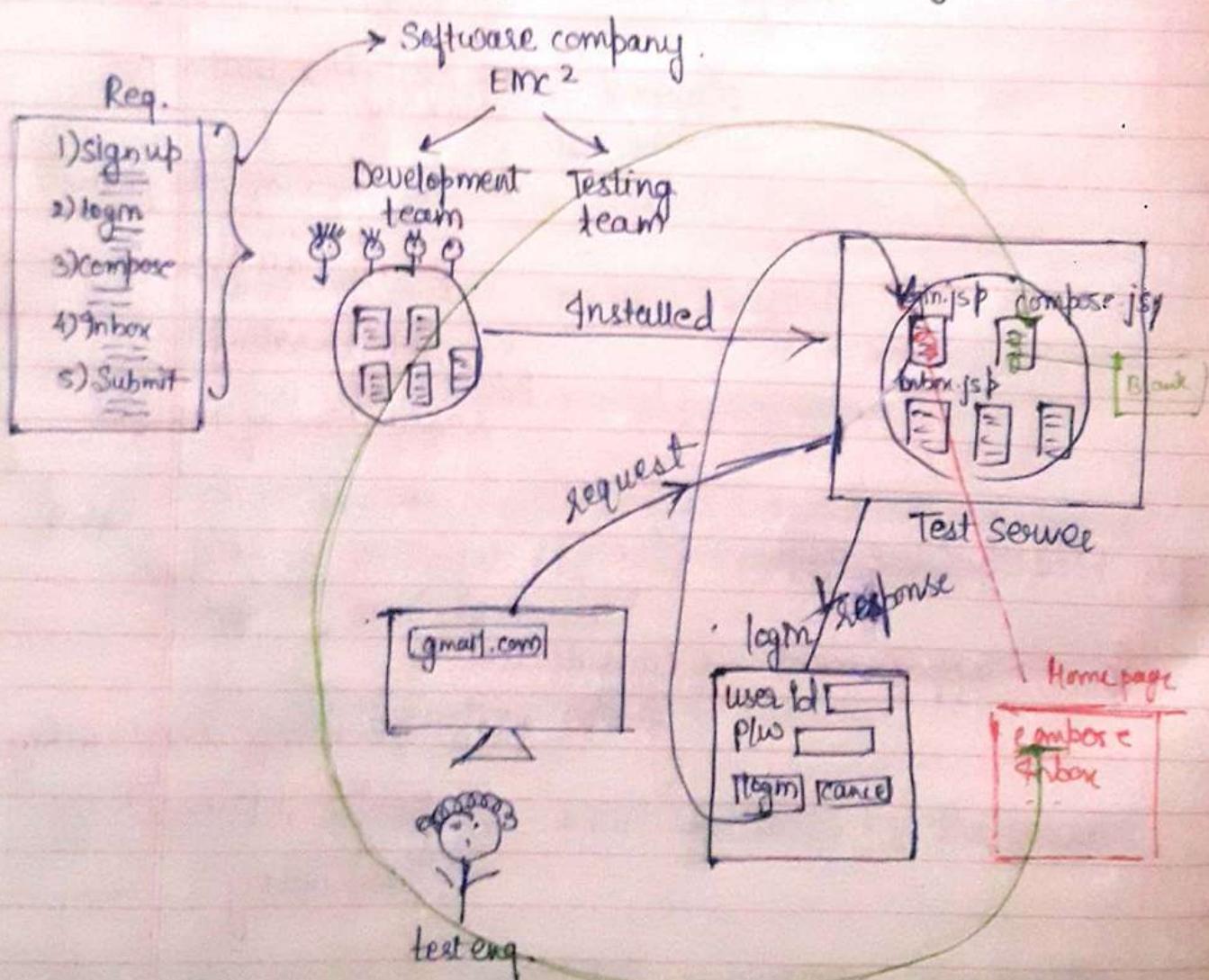
\* manual mistake can affect only 1, but form application  
many people will be affected if its wrong has error

### Eg 3. Defence:



### Definition 1:

The process of finding the defects in the software is called software testing.



### Definition 2:

The process of executing the program with intention of finding the defects is called software testing.

### Definition 3:

Checking the functionality of an application according to the customer requirement specification is called software testing.

## Types of Software testing

White Box testing

- (or) unit testing
- (or) structural testing
- (or) transparent testing
- (or) open box testing
- (or) glass box testing

Gray Box testing

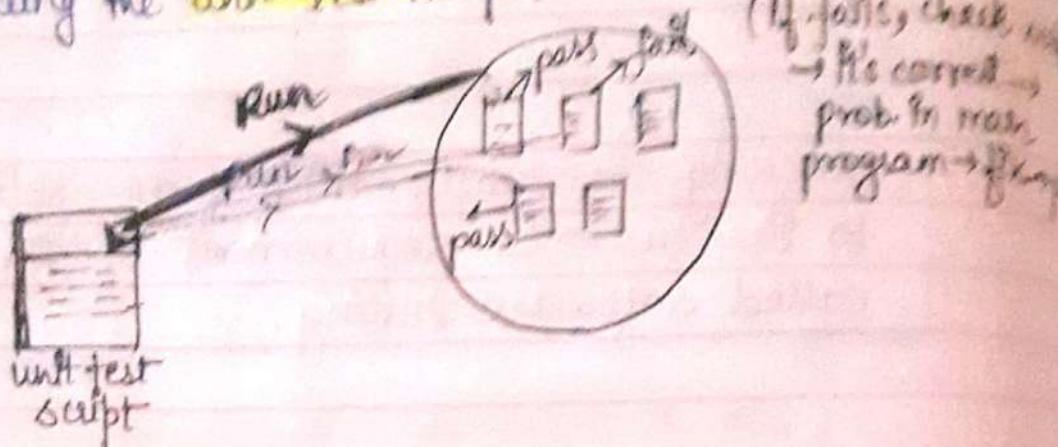
Black Box testing

- (or) closed box testing
- (or) functional testing
- (or) Behavioral testing.

### White Box testing

- \* Testing each & every line of the program is called white box testing
- \* It is generally done by the developer before giving the software to testing team.
- \* It is also called open box testing because source code is visible.

- \* It can be done manually & automatically also.
- \* We can do white box testing automatically by executing the unit test script.



### Black Box testing

- \* Testing the functionality/ behaviour of the application against customer requirement specification is called black box testing.
- \* Black box testing can also be done manually as well as automatically by using some automation tools like Selenium, QTP/UFT, Test complete etc.

Suppose developer have developed an application through which 1000 mails can be sent at a time, so in such cases doing manual testing will be tedious & here we go for automation testing.

Have you done white box testing?

In my project I know how developers did white box testing but I haven't done it, as I was busy with doing black box testing.

- \* We can do black box testing automatically by executing automation test scripts with the help of automation tools.

### WHITE BOX TESTING

- \* Testing each & every line of the program is called white box testing.
- \* It is generally done by developers & rarely done by test engineers, if they are good at programming.
- \* In order to do WBT, we should be good at programming.
- \* To do white box testing we should have knowledge of internal design of the code.
- \* Since source code is visible white box testing is also called as open box testing.

### BLACK BOX TESTING

- \* Testing the functionality of the application against requirement specification is called black box testing.
- \* It is done by test engineers.
- \* In order to do black box testing we need not to know programming.
- \* In order to do black box testing we need not to know internal design of the code.
- \* Since source is not visible in black box testing it is also called closed box testing.

## Black Box Testing

Testing the functionality of the application against the requirement specification is called black box testing.

### Types of Black Box testing :

1. Functional testing
2. Integration testing
3. System testing
4. Acceptance testing
5. Smoke testing
6. Exploratory testing
7. Globalization testing
  - L10N testing
  - I18N testing
8. Compatibility testing
9. Regression testing
10. Performance testing
11. Usability testing
12. Accessibility testing

## Functional Testing

Definition: Testing each & every component of an application thoroughly against requirement specification is called functional testing.

### Assignment 1

List GUI components for web based applications.

List GUI components for mobile applications.

List GUI components for stand-alone applications  
MS office, excel.

## Assignment 2

↓ login as admin

Add user :-	username <input type="text"/>
	password <input type="text"/>
Designation	<input type="button" value="Select"/> Test engg. Test lead Test manager
Phone No.	<input type="text"/>
Email ID	<input type="text"/>
Gender	<input type="radio"/> Male <input type="radio"/> Female
DOB	DD <input type="text"/> MM <input type="text"/> YYYY <input type="text"/>
Address	<input type="text"/>
<input type="button" value="Submit"/>	<input type="button" value="Cancel"/>

### SRS

Add user : on click Add user link, Add user page should be displayed & the following components should be displayed username text field, password text field ...

Username TF : - It should accept only 4-16 character  
- Special character should not accept.

Password TF : - It should accept only 8-24 character  
- It can accept digits & special character

Designation - Mandatory

phoneNo - It should accept valid 10 digit No.

Email ID - It should accept valid email ID

Gender - Mandatory

DOB - It should accept valid date

Address - It should accept 0-2000 character.

Submit - user should be created

Cancel - It should take to previous page.

#### Note:

We should always start testing the application by entering positive values. If the application is working for valid data than only we should test for invalid data. If the application is not working for one of the invalid data still we can test with some more invalid data.

# CBO - SRS

## 1. Welcome

1.1 login

1.1.1 username TF

1.1.2 password TF

1.1.3 login Button

1.1.4 cancel

1.2 forgot password

1.3 Remember password

## 2. Loan

2.1 PL

2.2 HL

2.3 VL

## 30 Amount Transfer

30.1 From account No. TF

30.1.1 It should accept 10 digit integer No.

30.1.2 It should be created by manager

30.2 FAN To account No TF

30.2.1 -

30.2.2 -

30.3 Amount TF :

30.3.1 It should accept only five integer b/w

100 - 50,000

30.3.2 It should not accept more than the balance.

30.4 Transfer Button

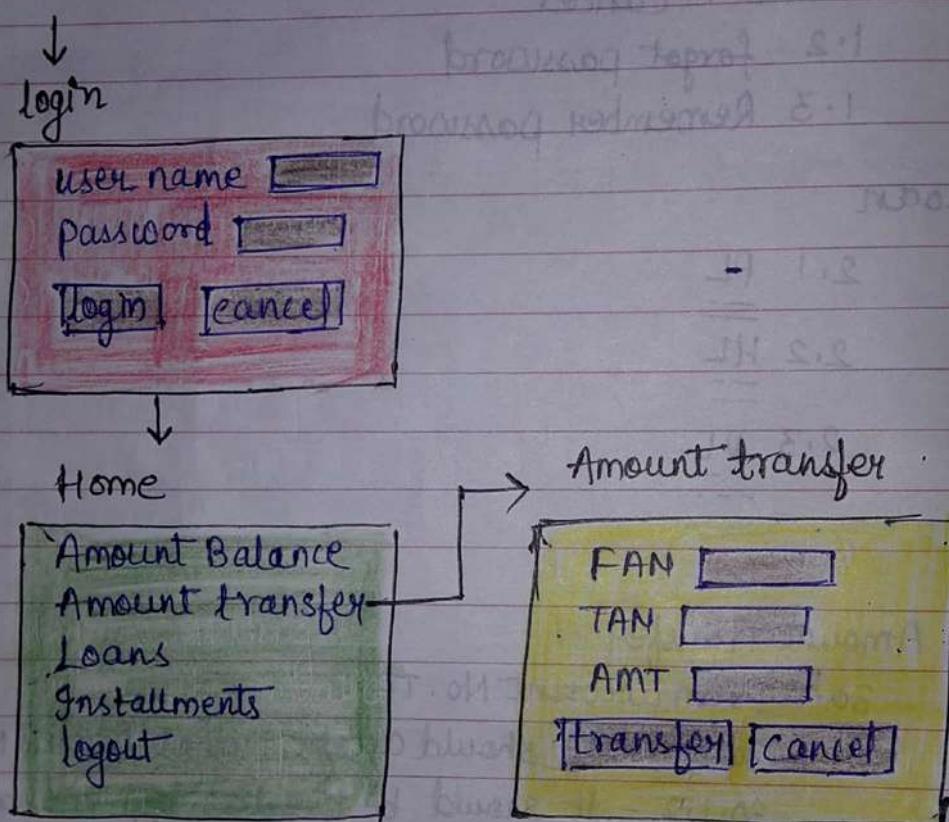
=

30.5 Cancel Button

=

Why the requirement document should be numbered?

- There will be clarity in the requirement & easy to understand.
- It becomes measurable.
- Easy to communicate with development team & customer.



Test scenarios:

①. FAN      10 digits ✓

9 digits ✗

11 digits ✗

alphanumeric ✗

blank ✗

space ✗

decimal ✗

Int. no. ✗

special character ✗

- 2). TAN - same as FAN  
 both account should be different.
- 3). AMT -
- 100 ✓
  - 90 ✗
  - 60,000 ✗
  - more than balance ✗
  - value ✗
  - decimal ✗
  - blank ✗
  - Space ✗
  - 0 ✗
  - 50,000 ✓
  - special character ✗
  - 1001 ✗
  - 10001 ✗
  - 99 ✗
  - 5000 + 100 ✗
  - 0100 ✓
  - $60,000 - 1000$  ✗ } no special character
  - $\frac{1000}{2}$  ✗
  - alphanumeric ✗
- wrong  
generations

Any kind of testing we can do in 3 ways.

- ① over testing,
- ② under testing
- ③ optimised testing.

Over Testing:

Testing the application with those scenarios which does not make sense is called over testing. By doing over testing we loose lot of testing time.

## Under Testing:

Testing the application with insufficient sets of data is called as under testing. By doing under testing we miss lot of defects.

## Optimised Testing:

Testing the application with those scenarios which really make sense is called optimised testing.

We have two types of testing

- ① Positive testing
- ② Negative testing.

### Positive testing:

Enter the valid or expected data & test the application is called as positive testing.

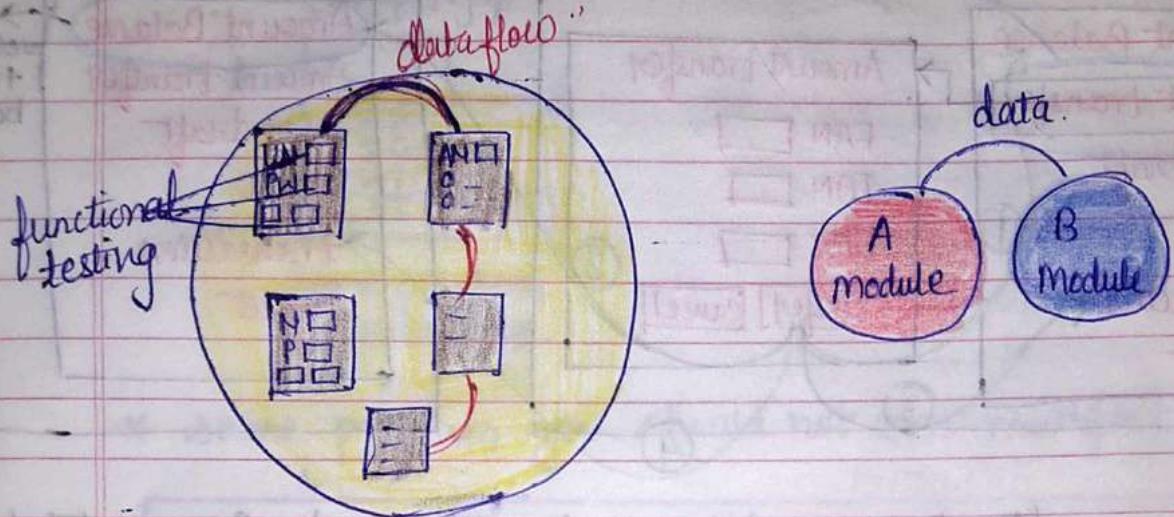
### Negative testing:

Enter the invalid or unexpected data & test the application is called as negative testing.

### Note:

As a test engg. we should not assume or propose the requirement.

# Integration Testing



Definition:

Testing the data flow between two dependent modules or features is called Integration testing.

Example:

- 1). login as a user A in facebook
- 2). sent request to user B
- 3). logout as A & login as user B
- 4). check whether request of A has come or not
- 5). Accept the request & logout B.
- 6). login as A & check whether friend request accept notification has come or not.

If these messages are received in both the accounts means there is data flow happening & this is just a small eg. for integration testing.

[ ]   
 Yes pass  
 No fail

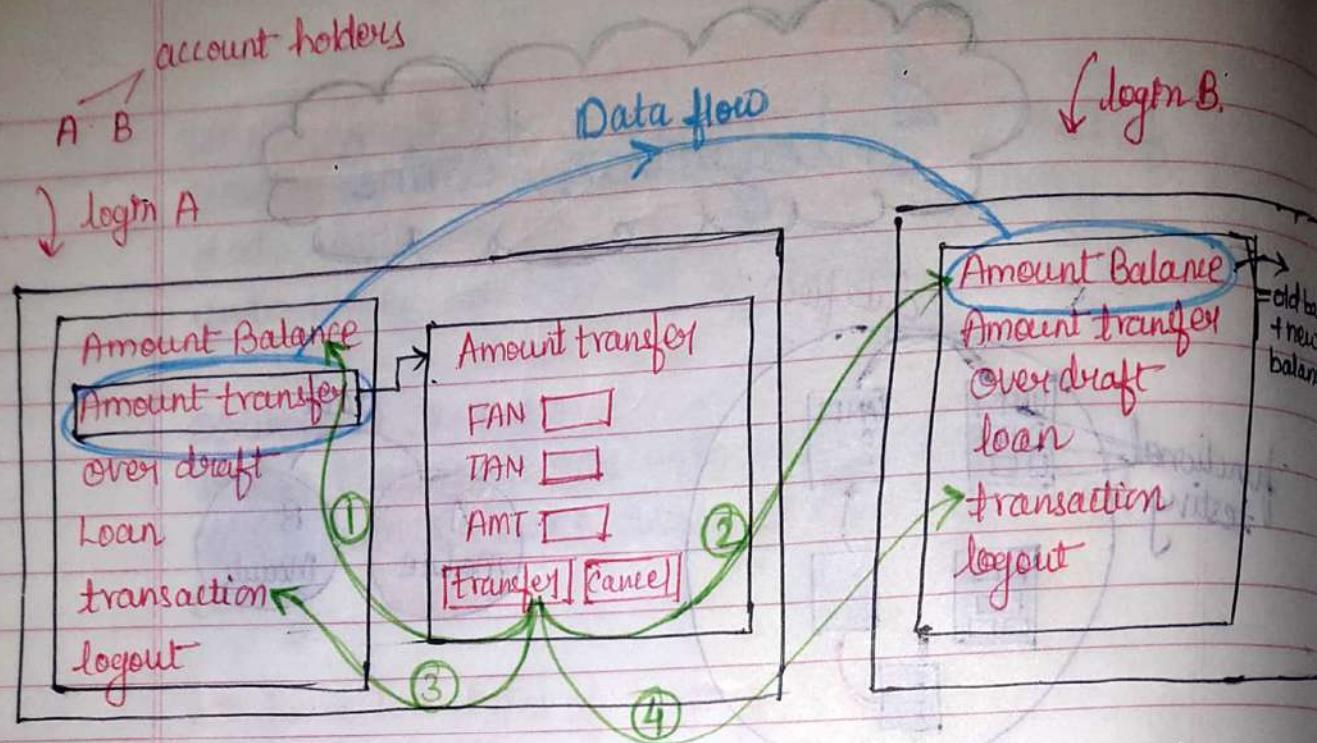


fig: Data flow between Amount transfer & amount balance

A transfers to B

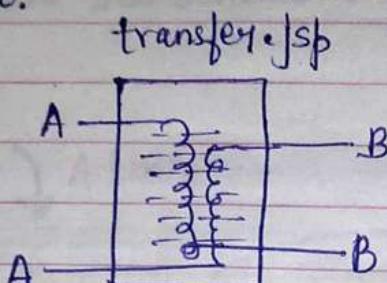
- Given
- 1). check B's balance
  - 2). check A's balance
  - 3). check transaction details of B
  - 4). check transaction details of A.

I      II (Best) → In less time, all  
      Given covered.

- Approaches:
- i) log in as A & do transfer
  - ii) log out as A
  - iii) log in as B & check transaction details.
  - iv) log out as B & log in as A.
  - v) check A's balance & transaction detail.

Is there any need to check that, If B transfer the amount to A, Is it working fine?

No.



Same program  
is executing.

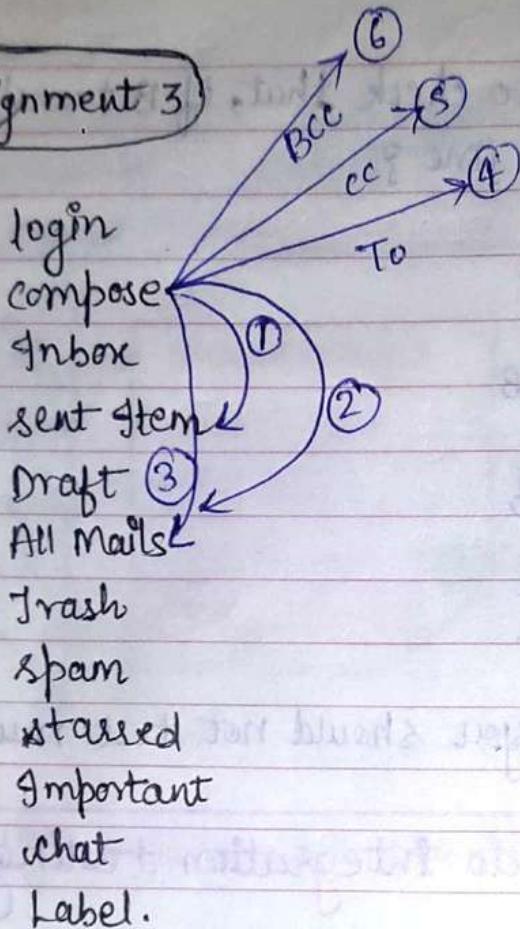
\* same program you should not test multiple times.

### Procedure to do Integration testing

- ① Understanding the application is very important. we should understand how each & every feature works in depth.
- ② we should also understand, how each & every features are related.
- ③ Identify all possible serious.
- ④ Prioritise the serious.
- ⑤ Document the serious.
- ⑥ Test the application by executing identified serious.

If the application is not working according to the serious & will be considering it as a defect & report to developers.

### Assignment 3



list all the scenarios related to compose feature.

#### Compose - Sent Item

- ①. Login to gmail as user A
- ②. click on compose link
- ③. enter valid data into all the fields
- ④. click on sent button
- ⑤. click on sent item link

#### Expected Result.

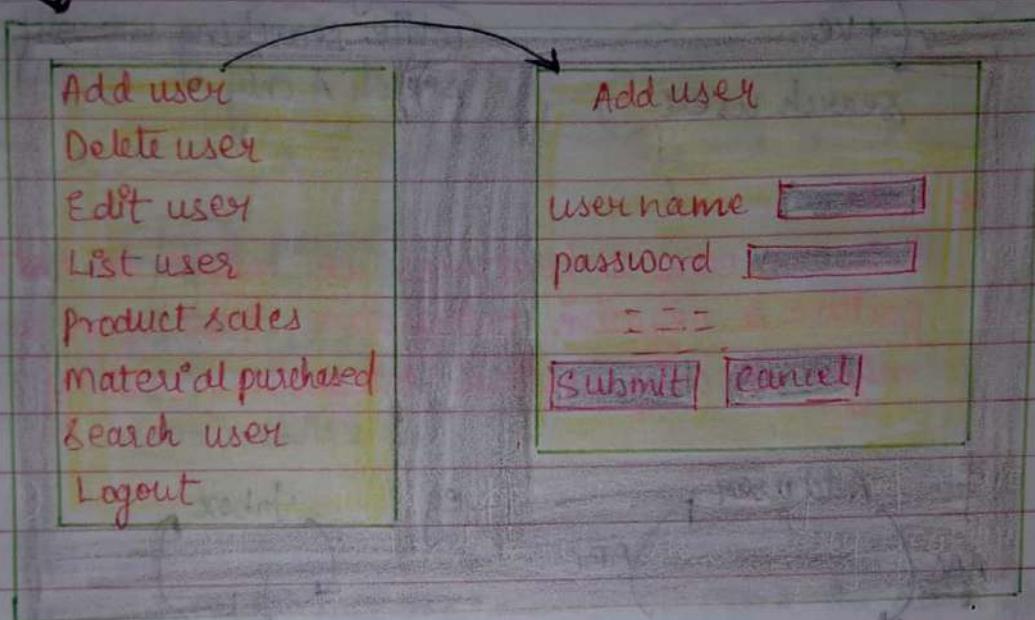
Sent mail should be listed in sent item page.

#### Actual result.

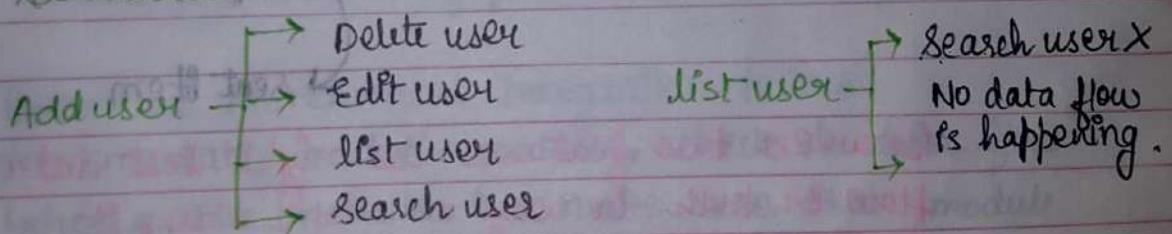
## Assignment 4

List 100 serious for gmail.

Admin



Serious



Delete user ->

list user

edit user

search user

add user

A diagram of a 'Delete user' form. It includes a field labeled 'username select' with a dropdown menu and a 'Delete' button.

Edit user ->

list user

search user

Delete user

Add user

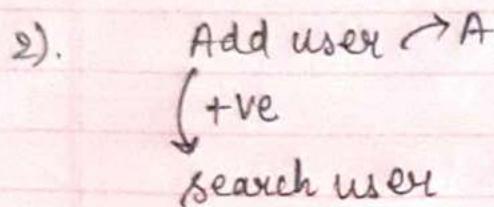
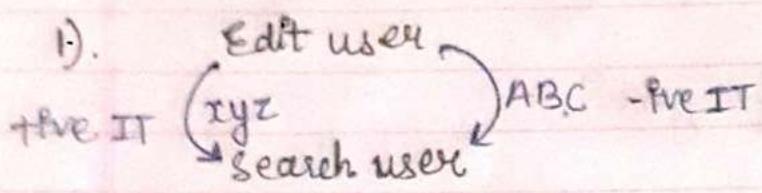
user name edited.

ABC → xyz  
2nd 1st

-ive IT +ive IT

\* Don't jump from feature to feature. stay on 1 find all the serious than move to next.

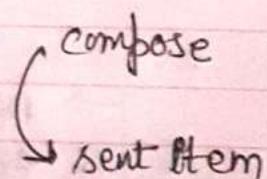
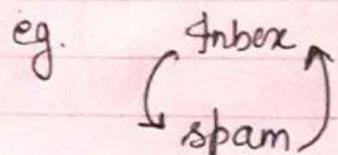
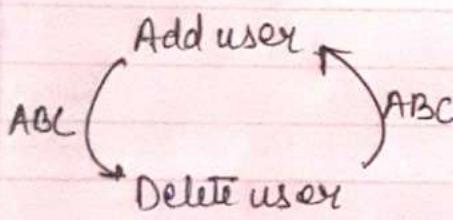
$ABC \rightarrow xyz$



If only A user is created,  
 while searching we will  
 search A only.

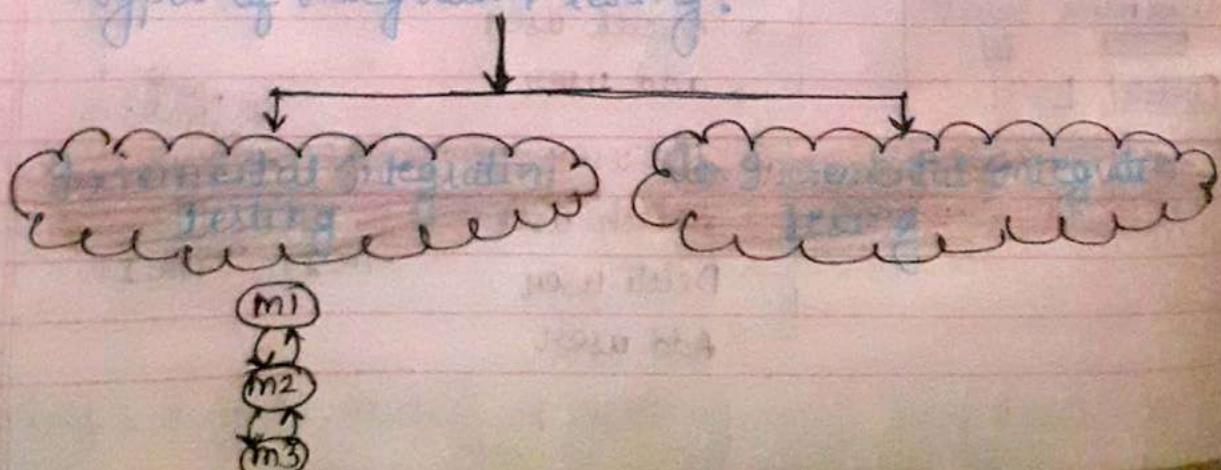
\* Note:

Between any 2 features we might do both  
 positive & negative integration testing or we  
 might do only positive integration testing.



Between two features we might test data  
 flow in both the way or only 1 way, It  
 varies from feature to features.

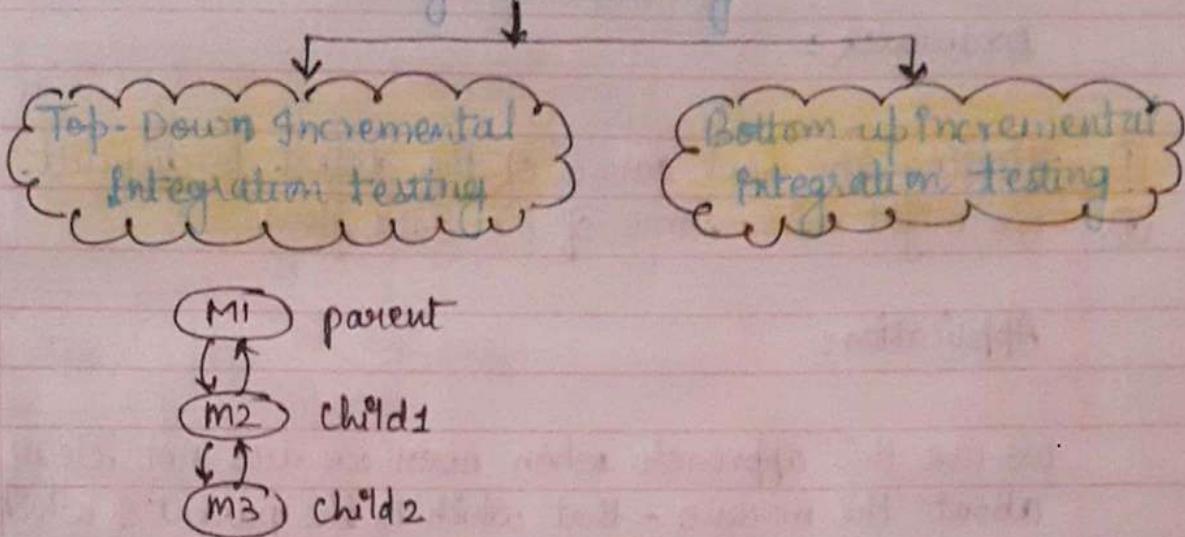
### Types of Integration testing:



## 1. Incremental Integration testing:

Incrementally add the modules & test data flow between the modules is called Incremental Integration testing.

### Incremental Integration testing



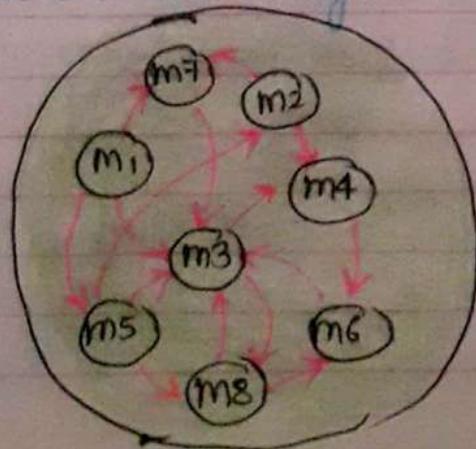
#### (i) Top-Down Incremental Integration testing:

Incrementally add the modules, check the data flow between the modules but make sure that newly added module is the child of previous module.

#### (ii) Bottom-up Incremental Integration testing:

Incrementally add the modules, check the data flow between the modules but make sure that module you have added is the parent of previous one.

## 2. Non-Incremental Integration testing: [Big Bang approach]



- \* Here we combine all the modules at a sout & check the data flow between the modules.
- \* It is also called big bang approach.

Drawback:

- ① Finding the root cause of the defect is difficult.
- ② We might miss some of the data flow.

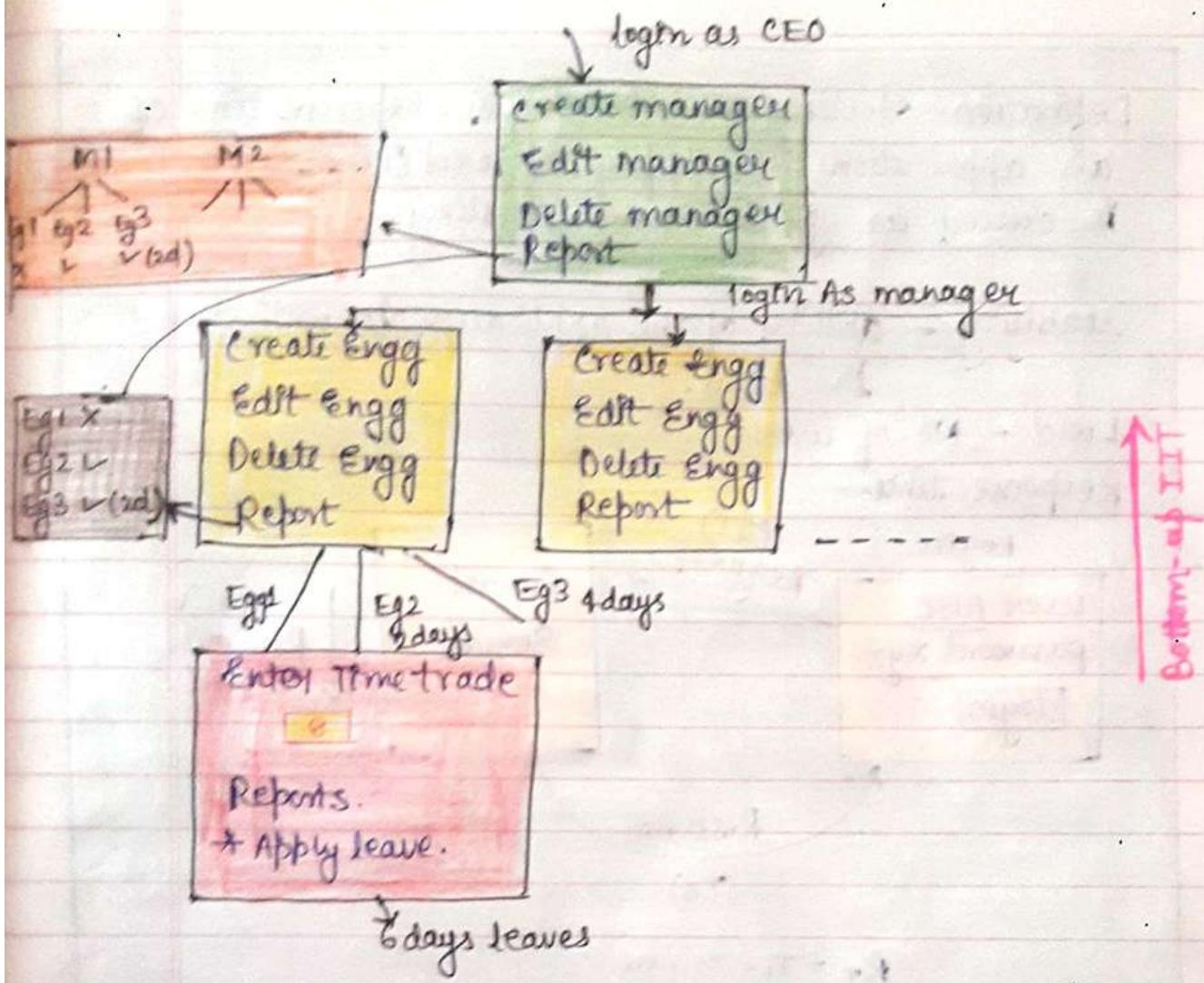
Application:

We use this approach whenever we are not clear about the module - that which is the parent & which is the child.

Ans.

- Definition IT
- Facebook eg
- Procedure
- Types.

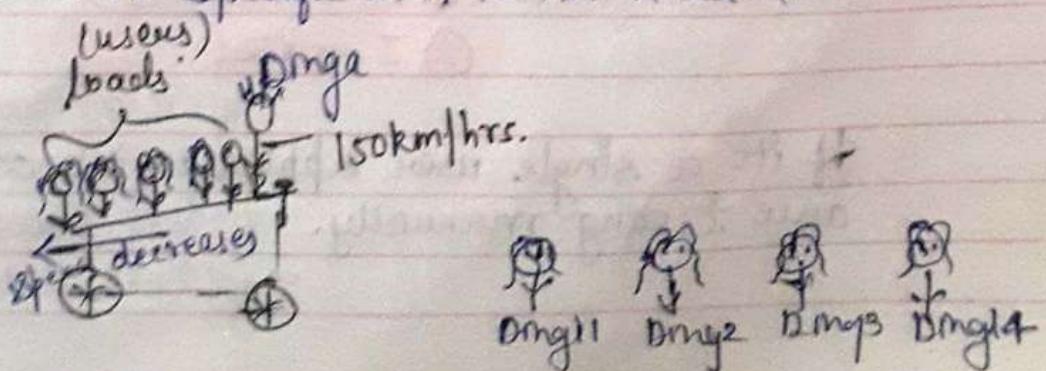
- \* When we can do both top-down as well as bottom-up testing it is called sandwich incremental Integration testing.



Eg: for Top-down, Bottom-up & sandwich  
incremental Integration testing

## Performance Testing

- \* **Non-functional / Scalability testing** - because here requirement specification is not needed.

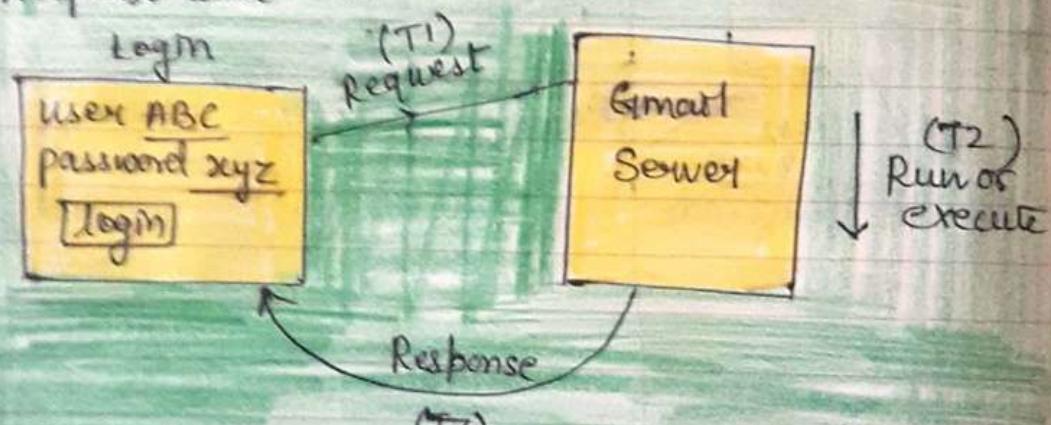


Definition: Testing the stability & response time of an application by applying load (no. of users) is called as performance testing.

Stability - ability of the application to withstand the load.

Load - No. of users

Response time -



$$R_T = T_1 + T_2 + T_3$$

## How to do performance testing?

For a multi-user application we can't do performance testing manually.

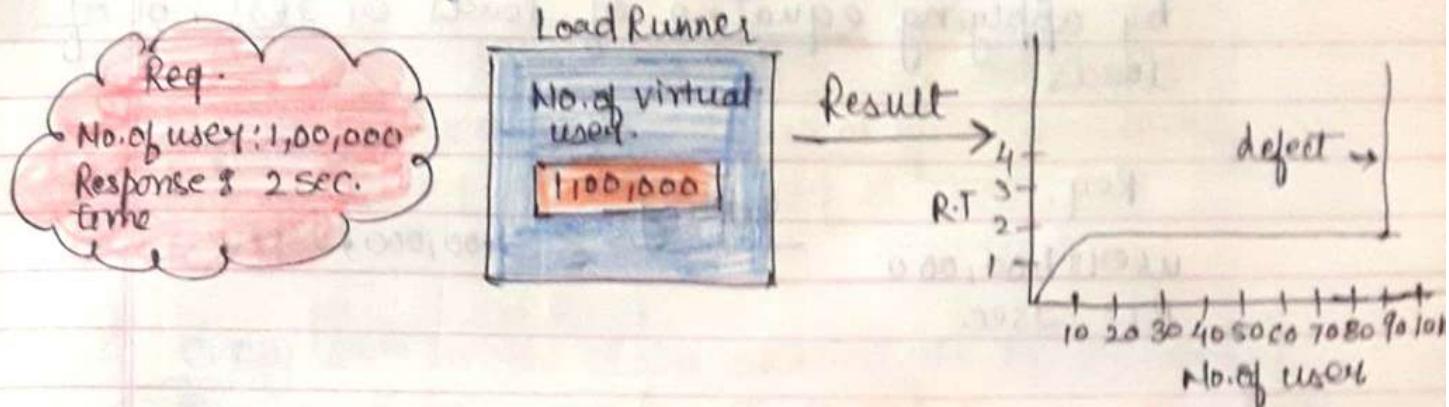
We always do Performance testing by using some automation tools -

- ① Load runner
- ② J-meter

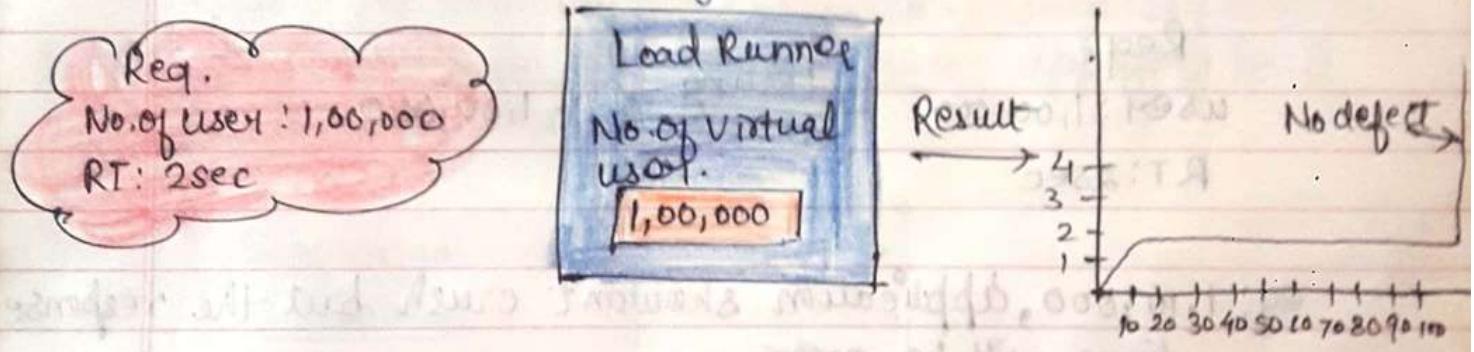
If it's a single user application we can do performance testing manually.

## How to do performance testing using load runner?

If it's a single user application



at 80,000 users the threshold value increased that means the response time of the application is high even before 1 lakh users. Hence it is a defect & we raise it as bug to Development team.



The expected result/actual result has to be with in 1 lakh user & response time has to be in 2sec.

After 1 Lakh users If the Response time ↑es it is not a defect

### Types of performance testing

- Load testing
- stress testing
- volume testing
- soak testing

## Load testing:

Testing the stability & Response time of an application by applying equal no. of loads or less no. of loads

Req.

user: 1,00,000  
RT: 2sec.

testing  $\rightarrow \leq 1,00,000, 2\text{ sec.}$

## Stress testing:

Testing the stability & Response time of an application by applying more no. of users is called as stress testing.

Req.

user: 1,00,000  
RT: 2sec

testing  $\rightarrow > 1,00,000$

eg: 1,10,000, application shouldn't crush but the response time will be more.

eg: 2,00,000, application shouldn't crush it will through an error.

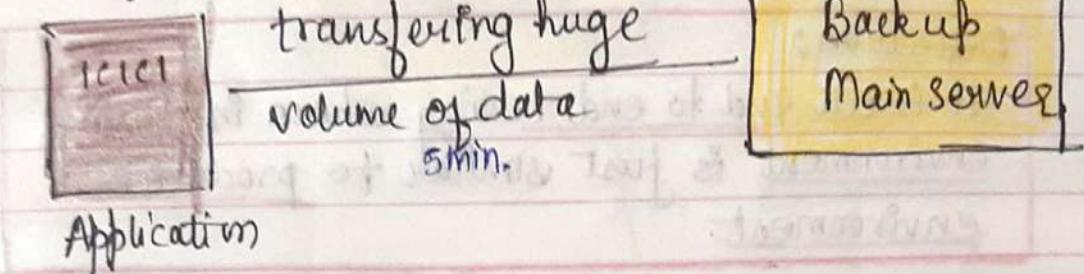
The application should not crush by applying more no. of users & if it crush it will be raised as a defect.

## Volume testing:

Testing the stability & response time of an application by transferring huge volume of data

from 1 place to another is called as volume testing.

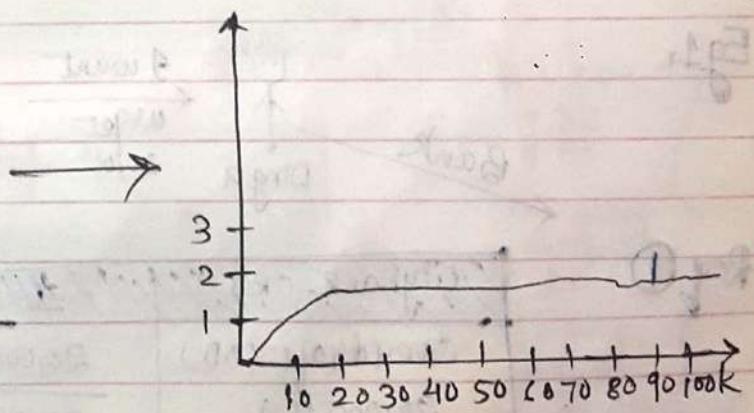
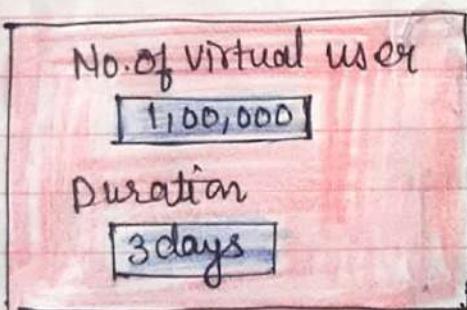
every 24<sup>th</sup> hr



every 24<sup>th</sup> hr. the data needed to be backed up  
it should take 5 mins if it takes more than that,  
then it's a defect.

## Soak Testing:

Testing the stability & response time of an application continuously for a period of time by applying load is called ~~of~~ Soak testing.



# System Testing

**Definition:**

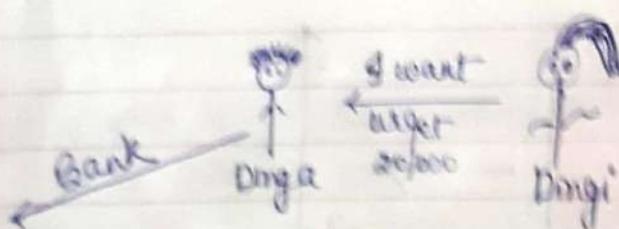
- \* It is an end to end testing where the environment is just similar to production environment.



**End-to-End testing :**

Navigate through all the features & check whether each feature is working or not.

Eg:-



Req. ①

Citybank-CR	
overdraft (OD)	20,000 (₹)
20,000 (₹)	20,000 (₹)
20,000 400 (2%)	20,000 400 (₹)
250 (AP)	
20,650	
- 20,650	
00,000	

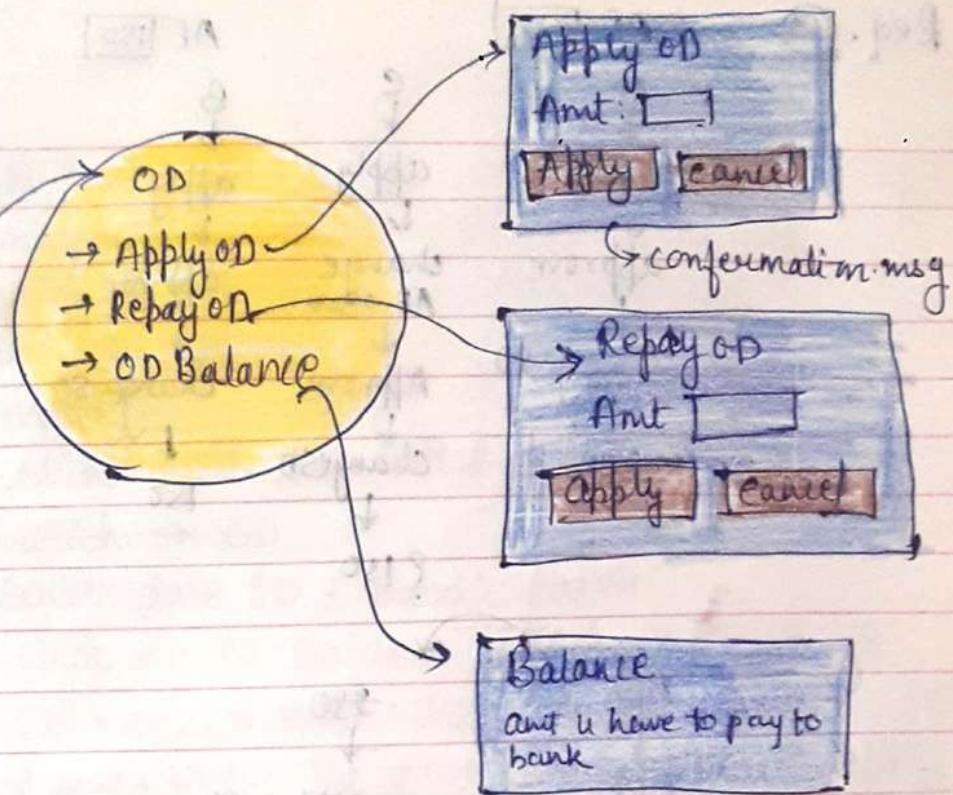
for 1 month

OD manager  
& Approve.

\* AP: activation fee.

wipro  
Dev. Test.  
log in customer

Amount Balance
Amount transfer
OD loans
Insurance
Logout



log in manager

Pending OD
Deposite
Logout
Create customer

Change AF  
AF [250] - 150  
Change Fwd

Pending OD
Name Amt Select
A 92,000 <input type="checkbox"/>
- - - - -
[Approve] [cancel]

Req ②

$A > 50K$   
 $+ \text{amt}$   
 $+ \text{AF}$

$A > \text{any}$   
 $+ \text{amt}$   
 $+ \text{AF}$

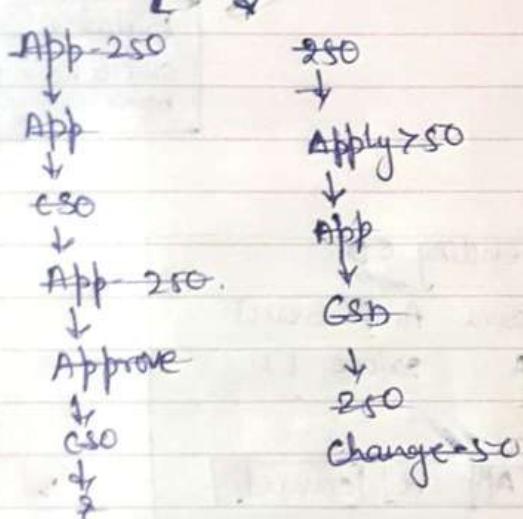
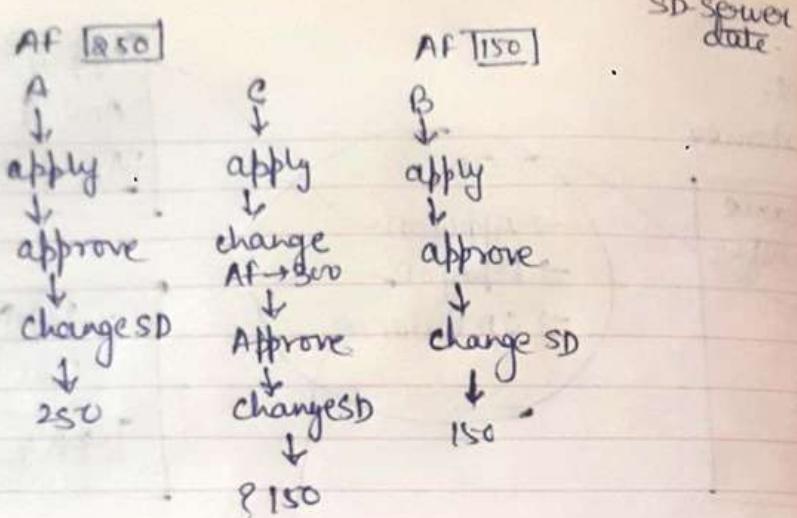
$A > \text{any}$   
 $+ \text{amt}$   
 $- \text{AF (Refund)}$

Deposit
Acc No. [ ]
Amnt [ ]
[Deposite] [cancel]

Name of Req:

Change activationfee: whenever the OD manager changes the activation fee, from that moment onwards if anybody apply for OD the latest activation fee should be charged.

Req. C



## Assignment - 5

Write the test sentences for all these 3 cases.

Steps for the 1<sup>st</sup> requirement:

- ① <sup>with</sup> login as manager.
  - ② create user A
  - ③ logout from manager.
  - ④ login as customer A
  - ⑤ click on OD & then apply for OD by entering amt = 20,000 & click on apply button.
  - ⑥ confirmation message should be received, that you have successfully applied for the OD of amt = 20,000.

- ⑦. logout from A
- ⑧. login as manager
- ⑨. click on pending OD
- ⑩. select A & click on approve
- ⑪. logout from manager
- ⑫. login as A & click OD Balance, it should be 20,000
- ⑬. login as A & click on OD
- ⑭. change the server date to 1 month later.
- ⑮. login as A & click on OD Balance which should be equal to  $20,000 + 400(\text{Interest}) + 250(\text{activation fee}) = 20,650$ .
- ⑯. click on OD repay, enter the amount & click on apply.
- ⑰. check OD balance it will be zero.
- ⑱. logout as A
- ⑲. login as A and click on OD then apply OD, enter the amount = 20,000 & click on apply.
- ⑳. logout as A
- ㉑. login as manager
- ㉒. click on pending OD, select A & click on approve.
- ㉓. logout as manager
- ㉔. change the server date
- ㉕. login as A & click on OD & then on OD Balance it should be  $20,000 + 400(\text{Interest}) = 20,400/-$   
if its the case, test is passed else its a deficit.

Eg2.

ICICI - CRS

Req1

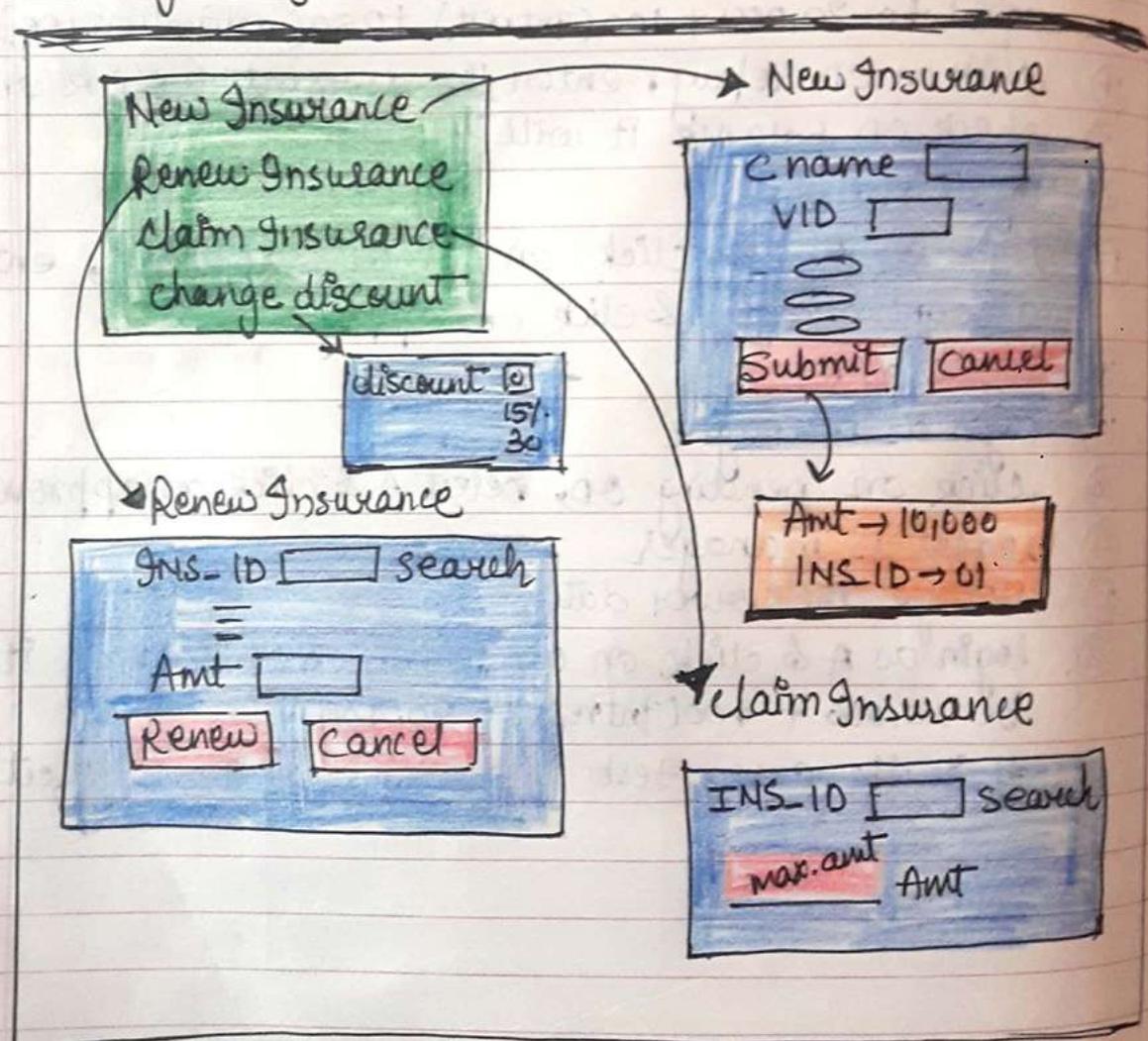
1y → 10,000  
2y → 10,000  
3y → 8,500 (15% discount)

Req2

If claimed: Node 2

wipro

Agent login



## Senarios for Req 1:

1. I will login as agent, click on new insurance.
2. New Insurance page will be displayed where I will enter customer name in C name field & will fill all other details & click on submit button.
3. Clicking on submit button I must get the confirmation message displaying amt = 10,000 & FNS-ID = 01.
4. Now I will logout as agent & will change the server date to next 1 year.
5. I will again login as agent and click on RenewInsurance option.
6. Renew Insurance page will be displayed where I will search INSID & it will display amt = 10,000 & then I will click on Renew button & send logout as agent.
7. Now I will again change the server date to next 1 year.
8. and will login as agent again.
9. Renew Insurance page will be display where I can search FNS-ID as 01 & amt displayed after it will be 8,500 in amt field.  
If this is the case, then test is passed else it has a bug.

## Senarios for Req 2:

1. I will login as an agent, click on new insurance.
2. New Insurance page will be displayed where I will enter all the fields & click on submit button.
3. Clicking on to submit button, I must receive the confirmation message where amt = 10,000 & FNS-ID = 01
4. Now I will logout as agent & change the server date to next 1 year.

5. Now I will again login as agent, will click on renew Insurance option. where I will get the Renew Insurance page where I can search the SNS-ID = 01 & amt must be displayed equal to 10,000.
6. Now I will click on claim option & I will get claim Insurance page, in which I can search Sns-ID as 01 and here maximum amount will be displayed.
7. Now I will logout as agent & change the server date to next 1 year.
8. And I will again login as agent and click on Renew Insurance option. where I will get Renew Insurance page where I can search SNS-ID as 01 and amt displayed in amt field must be equal to 10,000.

If this is the case then the test is passed else it's a bug & we must report it to the development team.

### Requirement 3.

when ever the Insurance agent changes the discount from that moment onwards if anybody creates new Insurance for them latest discount should be applied.

0

→ 15%.

→ A → 10,000

→ 30% B → 10,000

Change server date

Renew A → 10,000 B → 10,000

Change server date

→ Renew A → 8500 Renew B → 7000

\* 2 scenarios means 2 end-to-end testing

- from login to logout its 1 session.
- In interview give ex. of complex scenarios.

## Scenarios for Req3 :

1. I will login as a <sup>agent</sup> manager and will click on change discount option & there I will change discount = 15%. On next I will click on New Insurance, new Insurance page will be displayed where I will fill all the field for A & click on submit.
2. Now I will change click on change discount & will change it to 30% & will click on New Insurance, and fill all the field for B & click on submit.
3. Now I will logout as agent & change the sever date.
4. I will again login as agent & click on renew Insurance where I will get renew Insurance page, there I will search GNS\_ID for A and amount that should be displayed in Amt field should be equal to 10,000 &
5. I will again click with click on Renew button. This will renew the insurance for A for the 2nd year.
6. I will again click on renew Insurance, Renew option, renew Insurance page will be displayed, where I will search GNS\_ID for B and amount displayed in Amt field should be equal to 10,000 & then will click on Renew option. This will renew the B customer's Insurance for the 2nd year.
7. Now I will logout as agent & will change the sever date to next 1 year.
8. I will login as agent & repeat the same process to renew the Insurance for A & B for the 3rd year where amt field must display Amt = 8500 for A & Amt = 7000 for B.

If this is the case then test is passed else failed.

Assignment 6

write 1 eg. for system testing.

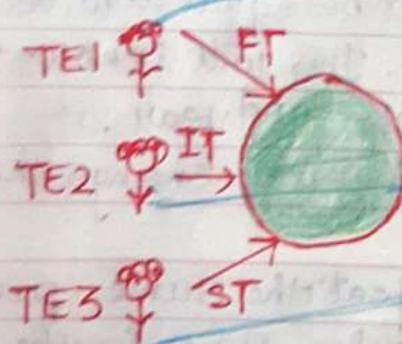
# Work allocation

Suppose we have an application. (e.g. gmail) & It has following features.

Sign up	Login	Compose
Inbox	All mail	Send items
spam	Draft	Trash
Important	Stared	Labels
Chats	Help	Settings
Outbox	Logout	Forgot P/w
Remember	Profile	Contacts
Calendar	Folder	Themes
Archive	Social	Promotion

Now I have to assign these features to different test engineers. (suppose TE1, TE2 & TE3)

Approach I



Approach II



Which approach is best?

Ans. II

Drawback of I approach

- ① Everybody has to understand the entire application

- ② TE1 may think it is a part of IT & will skip that part & TE2 may think it is a part of FT & may skip the same part. (confusing)

e.g:

Add user → A (create)



**Submit**

↳ confirmation msg

A (creating already existing user)

**Submit**

↳ creating A again.

This will be the bug left by them, rather than this it must show error msg that this user has already been created.

while allocating the work similar modules interlinked modules will be grouped together.

Compose

Inbox

Sent Item

Draft

All mails

Spam

Starred

Important

Archive

Sign up

Login

Logout

forgot p/w

remember p/w

settings

Logout

chat

calender

contacts

help

theme

folder

label

social

promotion.

(B)

(A)

more & main features.

(C)

not interdependent much

Dividing the work in this way, may also lead to more work for 1 group & less for another, but still we will go this way.



→ A

TE1: Dinga (4+ years exp, talented (but he escapes), follow up is required)



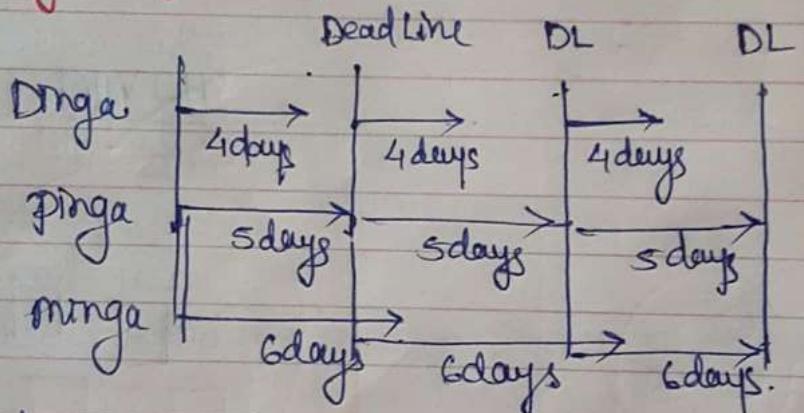
→ B

TE2: Pinga (1 year exp, humble, workhard, trusted)



→ C

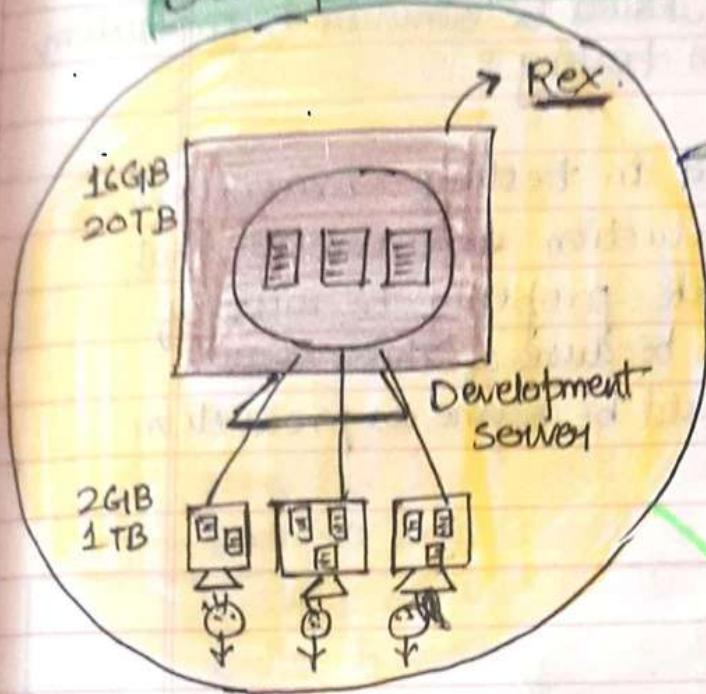
TE3: Minga (fresher)



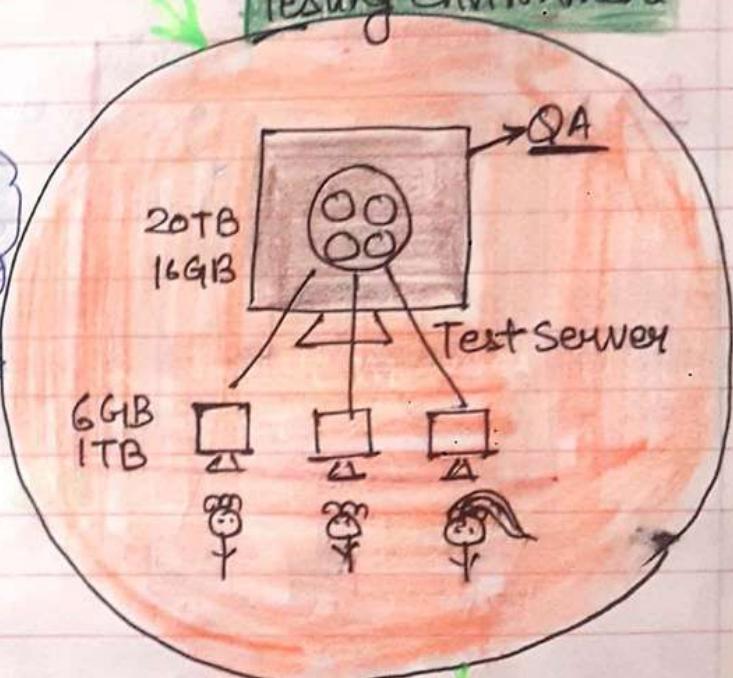
Lead will keep on observing, he will not instantly react, when he see that Minga gets late for the 3rd DL also meet the works regularly mean he really has lot of work & then he will adjust for it.

He comes to know that Pinga is really working fine & is talented also he will ask Dinga - that Dinga you please do Pinga's work he is not working fine & will give the assign Dinga's work (critical section) to Pinga as he don't require any followup.

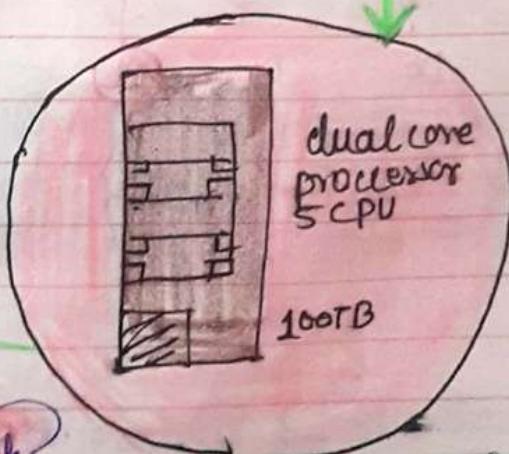
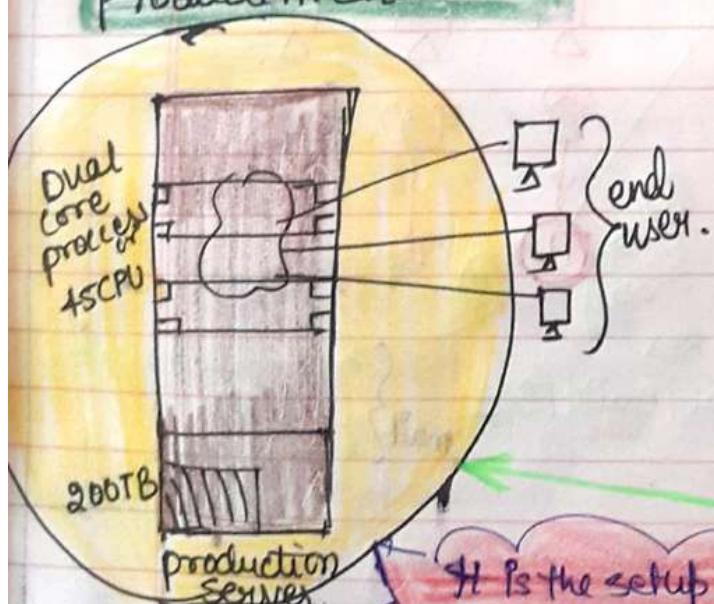
## Development Environment



## Testing Environment



## Production Environment

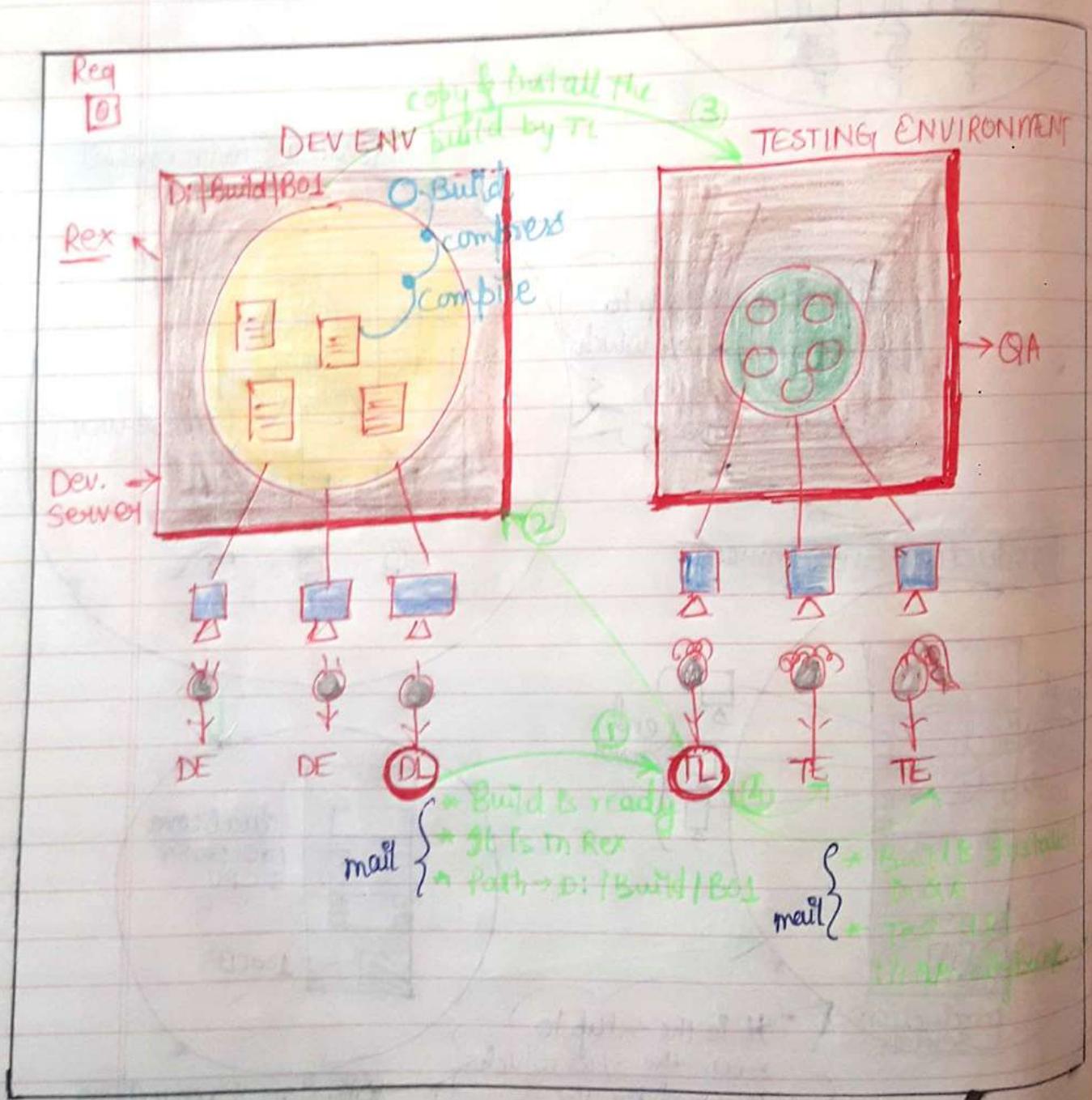


A pink cloud-like callout bubble to the left says: "It is the setup to run the SW which is used for the business."

## Similar Production Environment

why testing environment should be similar to production environment in system testing?

The software we tested in testing environment & installed in the production environment that software may not work properly because of different configuration because of this reason testing environment should be same as production environment.



- \* Developers will develop the program & will compile & compress it to create the build.
- \* Once the build is created it will be stored in some drive & Development lead will inform to the Test lead that the build is ready & stored in Rex server <sup>in some drive</sup> name) & link for its path will be given in a mail.
- \* Now test lead will click on the path given & will login to Rex server & copy the build & install to testing server & will inform to Test engineers that build is ready & will provide the link for path to them through mail.
- \* Now TE's will test the build according to the modules assigned to them.
- \* If test er. finds the defect he will inform to Development lead & then he will inform to the related development engineer & the DE will fix it.
- \* As soon as you get the defect, you must immediately raise it.  
(if not - u may forget it, anybody else may take the credit), developer's time will be wasted if report is given at EOD)
- \* As a test lead if you get the new build then, you must uninstall the previous one & install the new build in your test server for further testing the application.
- \* Suppose 100 defects has been raised by the testing

team & development team have fixed 60 defects out of it & have sent the new build, then we will have to retest only the fixed part & will never retest unfixed part.

- \* When ever we find the defect we make a report through some tools & a no. is also given to that bug which give ease in identification & understanding
- \* Defect can be raised by any one (any tester) but the one who raised the defect, & will only be doing the retesting for it.
- \* Author of the feature should test the feature, author of the defect should retest the defect.
- \* There will be some features like login, sign in which will be used by all but author will be only one.

Commonly used  
Signup  
login

forgot p/w  
remember p/w

;

;

compose  
sent item

Inbox  
Draft

;

Calander  
profile

contacts  
chat

;

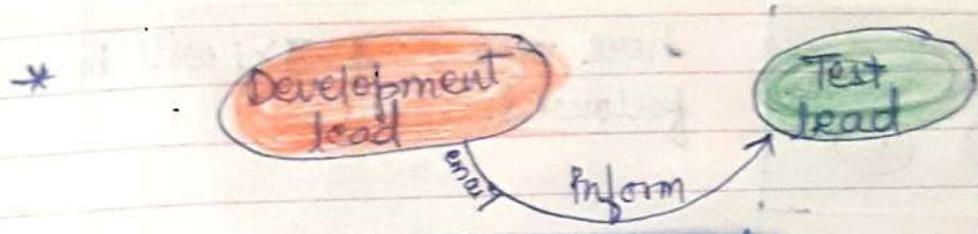
TE :- Dingi

TE :- Dingi

TE :- Ningi

defect in login was raised by Dingi, because that was common. but the features those are not common will not be raised defect

by others because they will be busy in their own modules only & others can take revenge of it by doing the same. But anybody can raise the defect any time.



- \* **Buit is ready**
- \* path → D:\Builds\B01  
B02

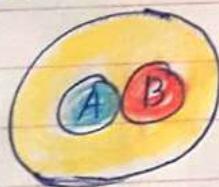
\* module A is developed  
(OR) \* module B is developed  
so defects of A is fixed.

D1 -- ?  
D2 -- } Details about  
D3 -- } fixed defects.

- \* Build: B01.



Build: B02

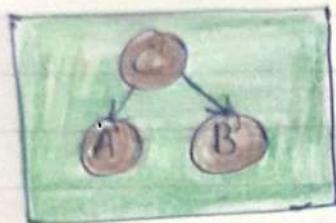


Build02 will be having containing the new module as well as the previous one also.

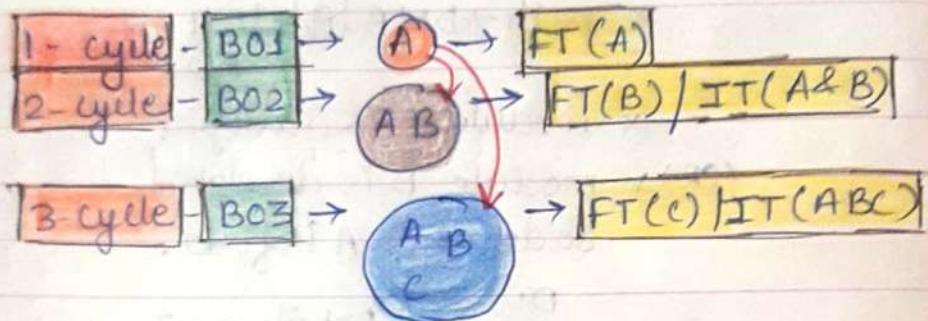
- \* ① Test old module A
- ② Test New module B
- ③ Defect fixed in A
- ④ Integration b/w A & B

$$[2 \geq 4 \geq 1]$$

If you follow this strategy, you can catch the defect soon, but its not the international standard to be followed.  
we will first test the new module, because manager will ask the status for the new modules first.

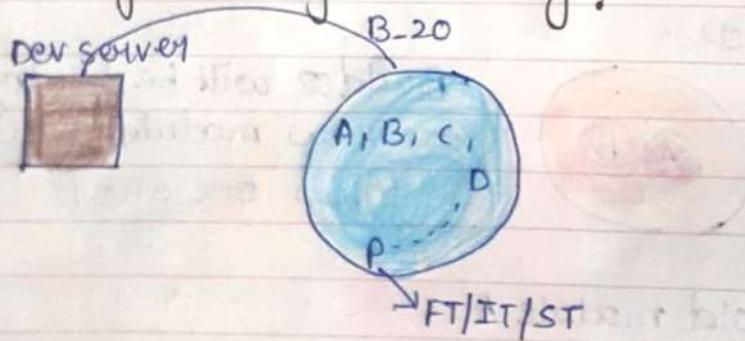


here also  $2 > 4 > 3 > 1$  will be followed.



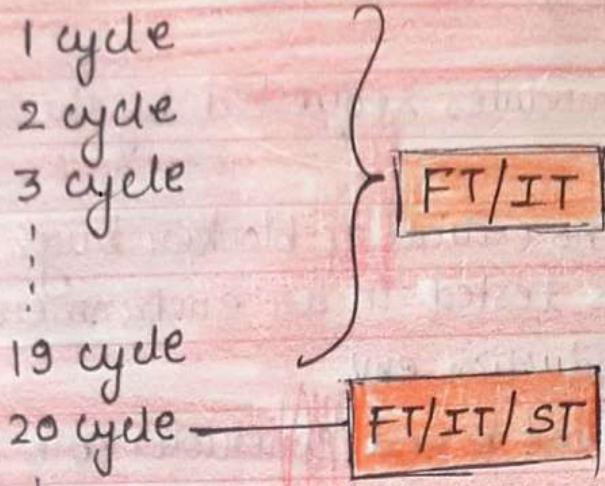
Defects found in the first test cycle might be fixed in the second test cycle or in third test cycle or in forth or at the end of the release depending on the priority of the defect.

Q. Where you do system testing?



- ① Minimum bunch of features should be ready.
- ② Testing environment should be similar to production environment, is ready.
- ③ Old / basic features should be stable.

Up till add to cart features its application is ready but its not sufficient to do online shopping because payment gateway is not added.



40 cycle ————— B-40 A, B, C  
Z

all the modules requested by the customer are delivered.

41 cycle  
42 cycle  
43 cycle  
44 cycle  
45 cycle

last few cycles are for testing, no new module is added in FT.

A, B, C  
Z

RETESTING  
(testing whether the defects are fixed or not.)

46 cycle

No defect, all modules are beautifully working, all business scenarios are working fine, no block & critical defects, s/w is working fine in testing environment similar to production env.

⇒ It's the time to release the s/w

S/w can have many release (e.g. whatsapp ask for the update with its new release).

Q. When do we go for release?

- (i) When all the modules requested by the customer are ready.
- (ii) When there is no critical or blocker bugs.
- (iii) When the s/w is tested in an environment which is similar to production env.
- (iv) When all the end to end scenarios are working fine.

Q. What is release?

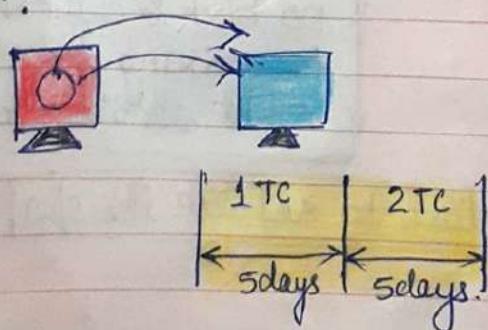
Starting from gathering requirement, develop the s/w & test it for so many cycles till until you launch the s/w to production is called as one release.

Q. What is test cycle?

It is an effort/duration spent to start & complete the testing is called test cycle.

1 test cycle might take 3 days, 4 days, 5 days, 6 days  
It depends on size of the application, complexity of the application & no. of test engineer.

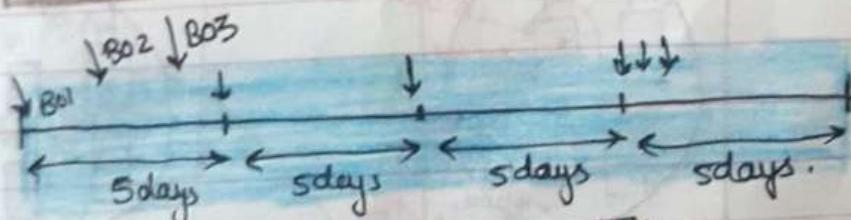
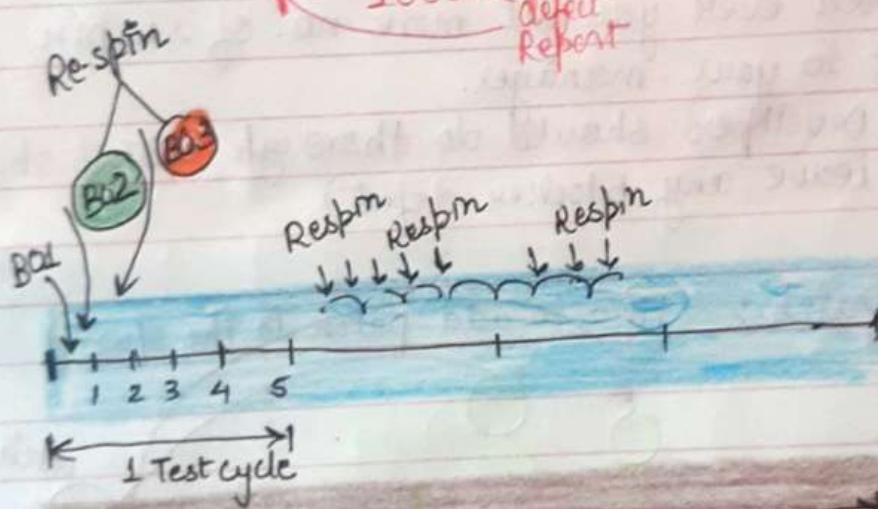
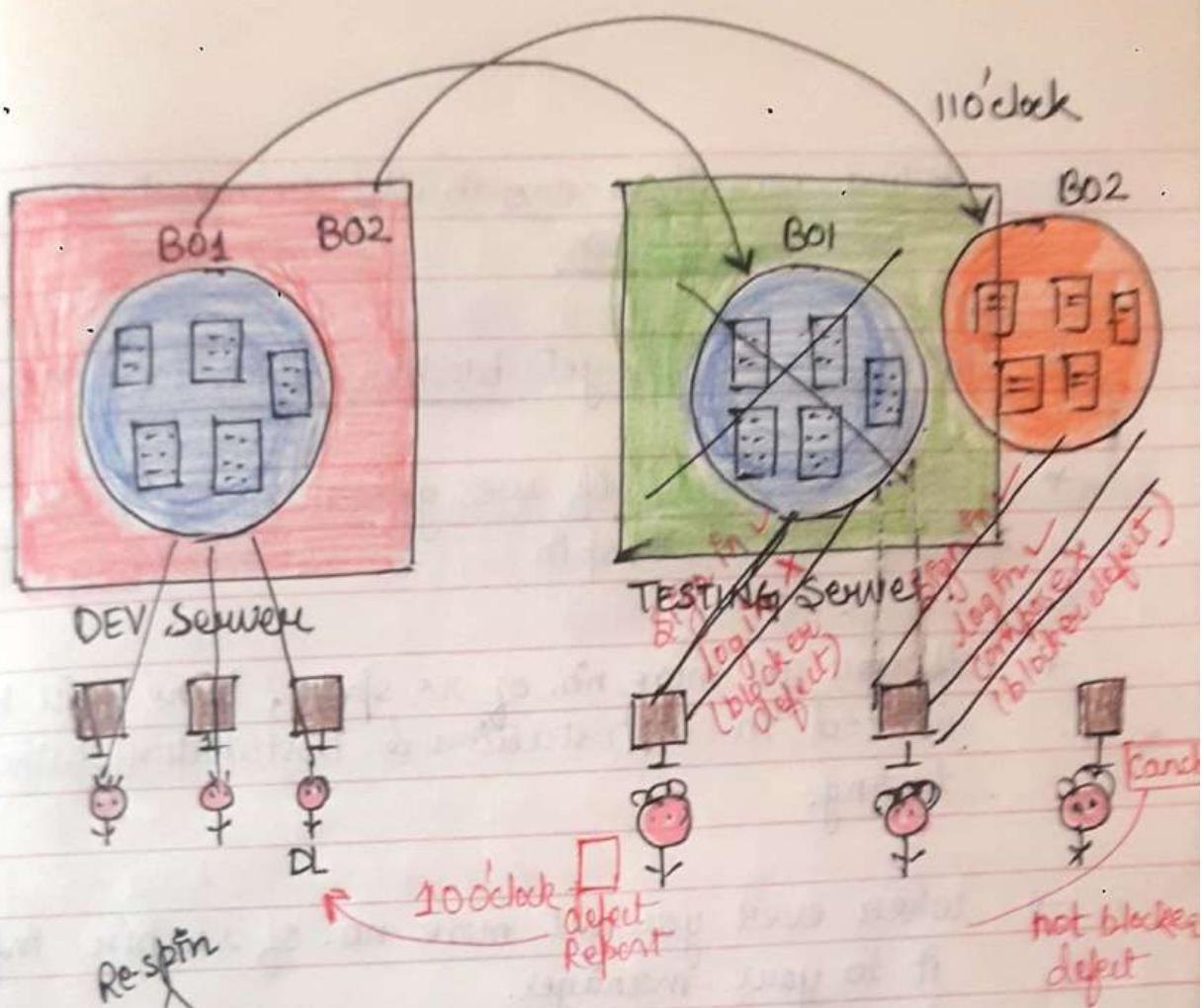
Q. In the current project how many cycles you have tested?



1 year Experience

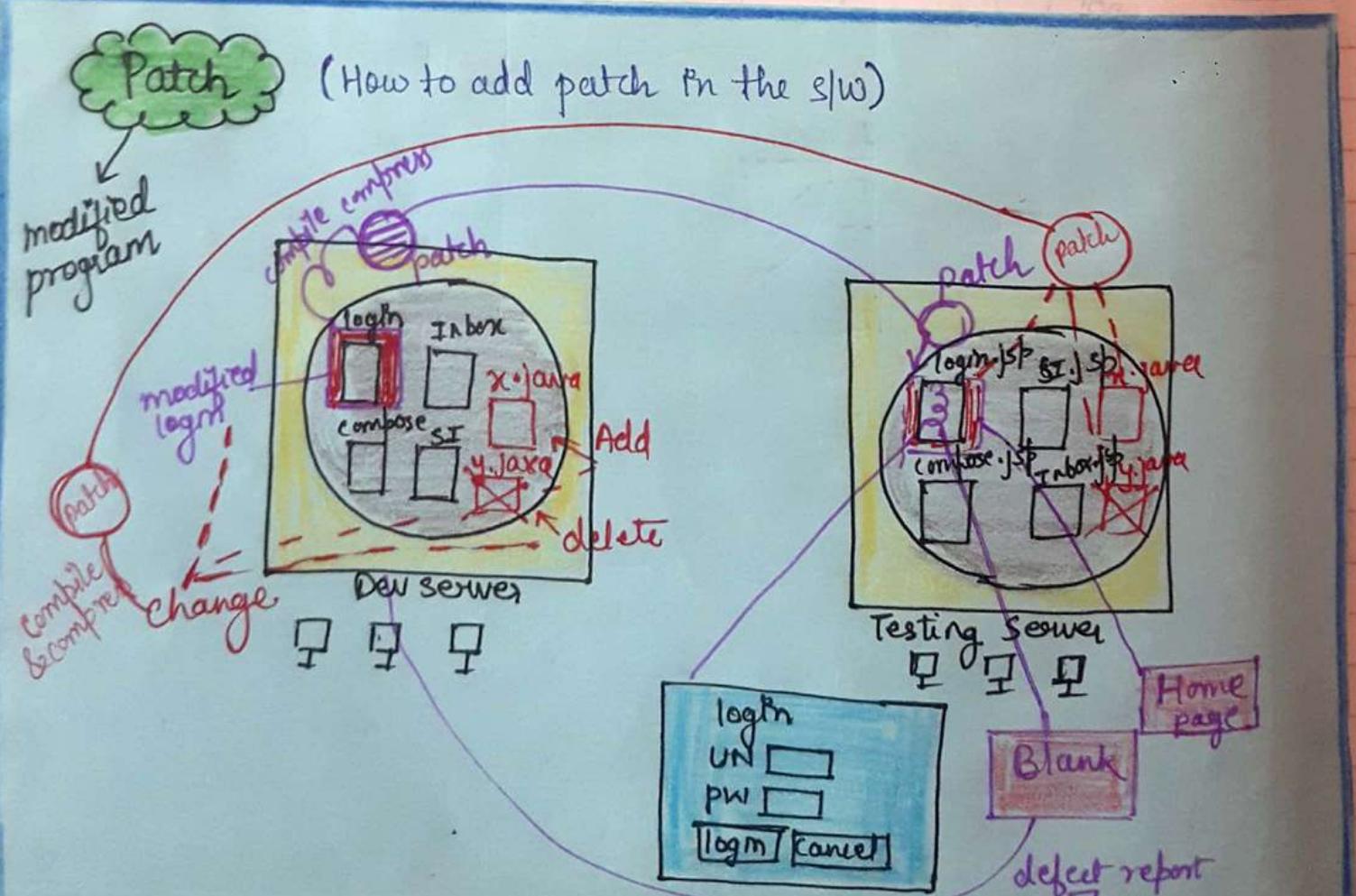
$$\begin{aligned}1 \text{ month} &= 4 \text{ cycles} \\1 \text{ year} &= 4 \times 12 \\&= 48 \text{ Testcycles}\end{aligned}$$

(48-50 TC say)  
• Show like you are thinking & answering.

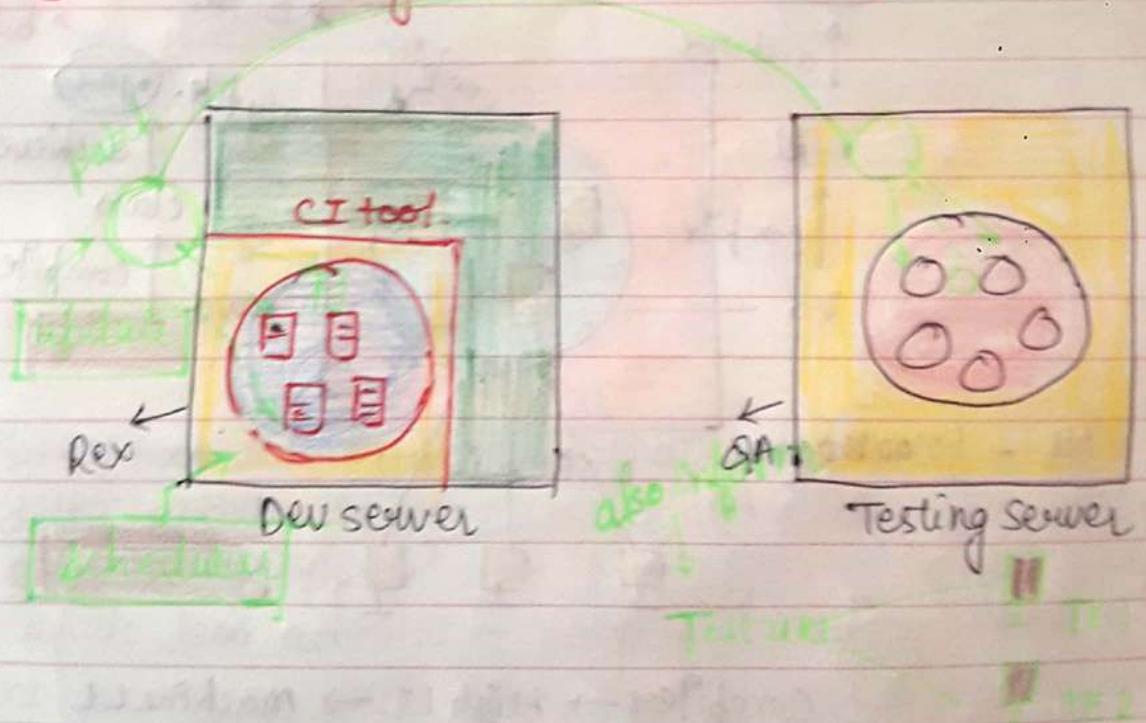


  
map to  
manage  
about many  
re-spin

- \* Getting more than one build in a test cycle is called as Re-spin
- \* Whenever we get blocker defect we get Re-spin.
- \* In a test cycle if we get 'n' no. of builds also, it is called re-spin.
- \* If you get more no. of re-spin, time will be wasted in uninstallation & installation rather than testing.
- \* Whenever you get more no. of re-spin inform it to your manager.  
(Developer should do thorough WBT & should not leave any blocker defect).



- \* patch Is a modified program, whenever you get patch you don't need to uninstall you just have to update. you will get patch whenever there is a small defect or blocker defect.
- \* Patch will be made only when there are some small changes else build is created, if you get new build you need to uninstall & reinstall it again.
- \* changes in program can be like -
  - adding a new program
  - modify the existing one
  - or deleting the existing one.
- \* Continuous Integration tool: (CI tool)

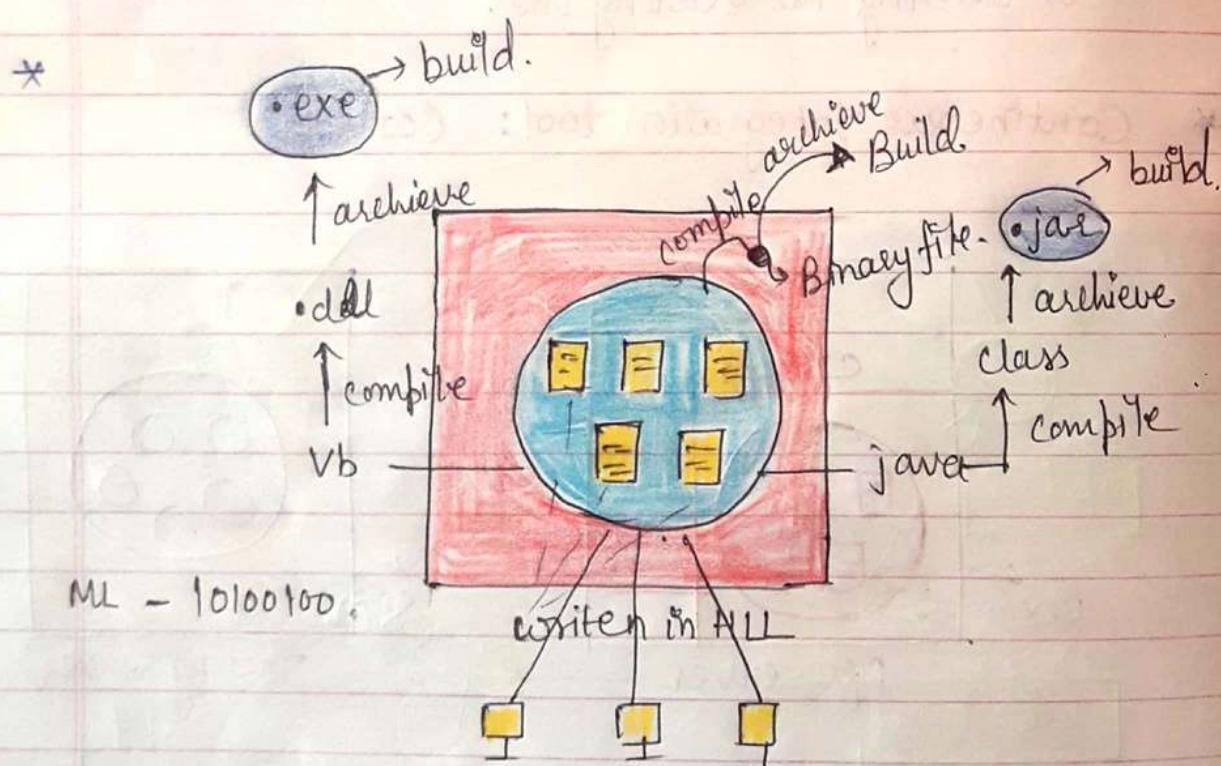


CI tool is installed in the development environment

update button is present in the CI tool. while developer click on update it automatically picks all the

updates & make a patch of it & sent to the test server & install it there & also update (inform) the test engineers that the patch is ready along with test path.

- \* continuous integration is done by CI tool between dev & test server team, so it is called continuous integration tool.
- \* scheduler feature is present in CI tool through which it automatically take update in a fixed interval of time even without clicking on update.



Compiler → High LL → machine LL

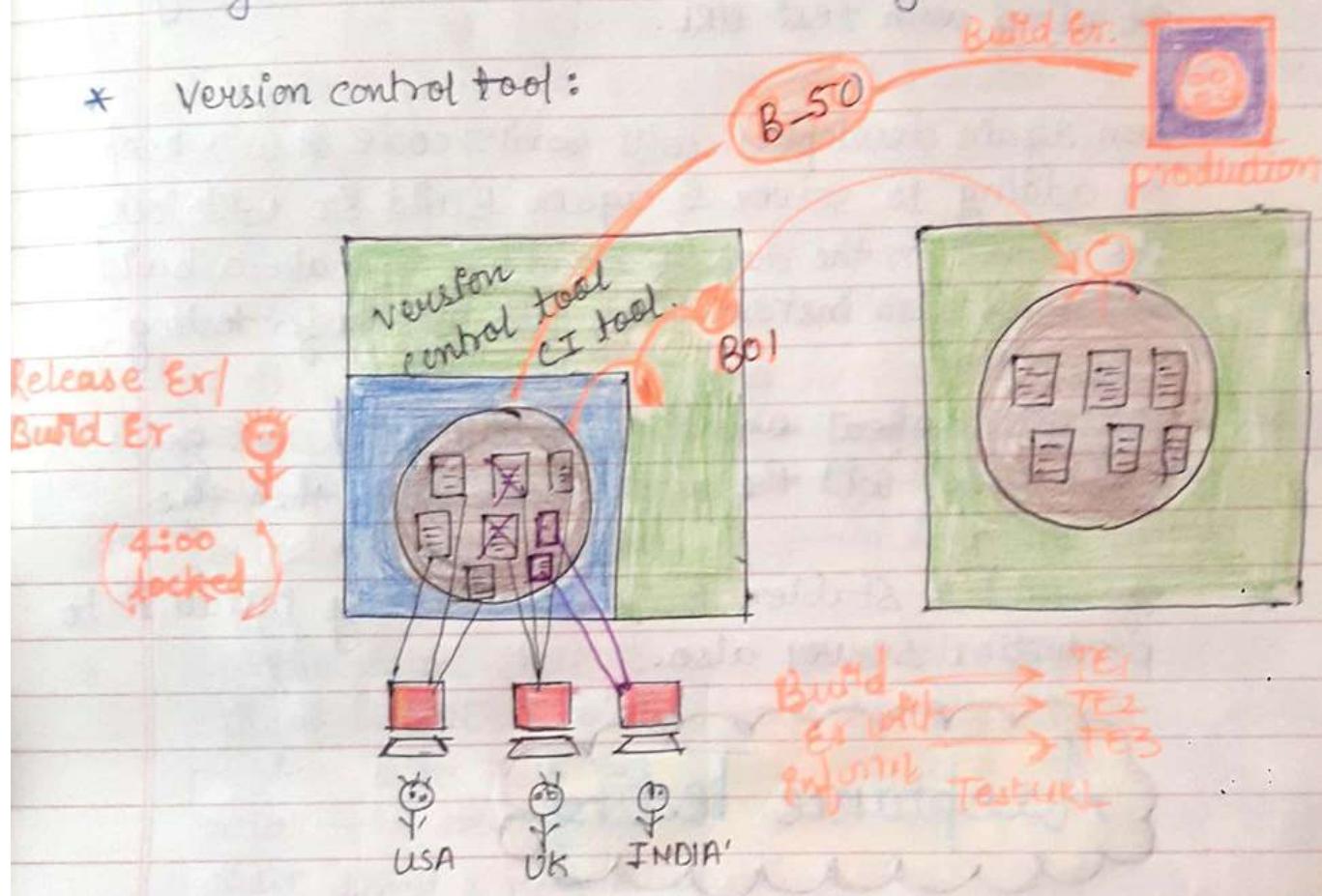
error → compile time error → syntax mistake

Execution → runtime error → logical mistake

Q. Who is involved in the installation of the SW?

Anybody from testing team or anybody from development team or release engineer or a field engineer, might be involved in installing the SW.

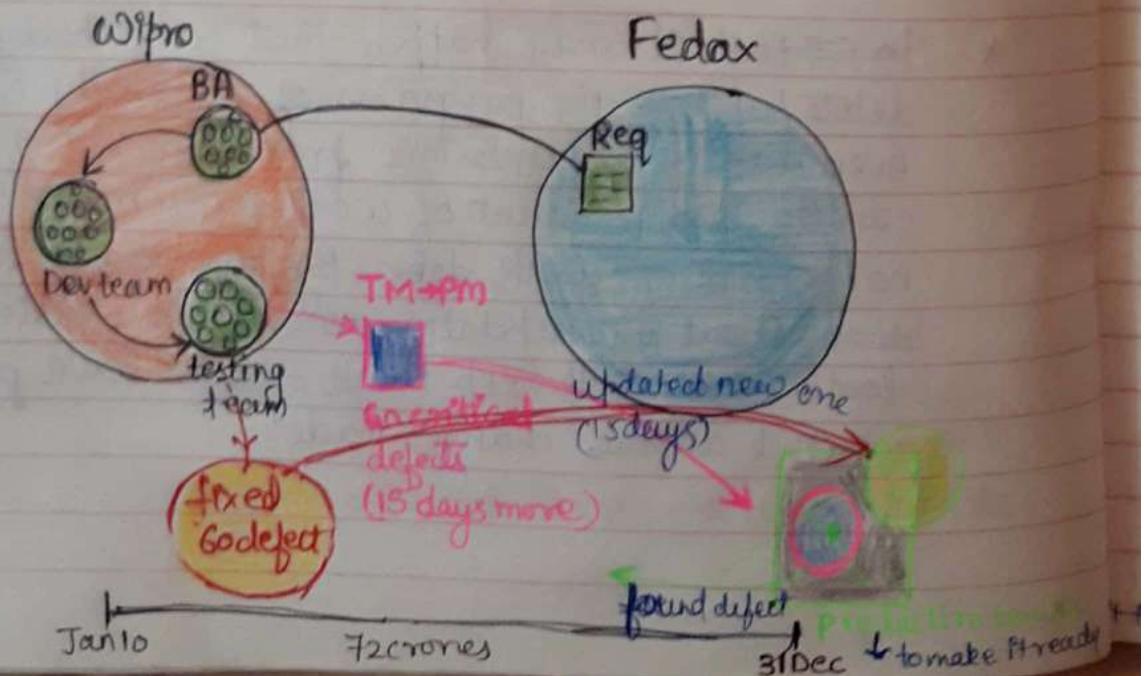
\* Version control tool:



\* In CI tool it may happen that some body might delete/update the programs of somebody else, without even taking permission from him or some body can do less amount of work & go, mean there is no record of work done by everyone & no. information about the updates. So we use version control tool - which will keep track of who have performed what & all the changes made.

- \* We have build Er. or we call release Er. who informs to the development that at some hr. of the day he will lock the development server & then he will create the build & install it in testing server & will also inform all the testing Er. along with test URL.
- \* Then again developers will write code & will keep on adding to server & again Build Er. will lock the server on the day of deadline & make a build & install also install it on test server for testing.
- \* This will repeat under the build until we get a build which will be the most stable one, then the
- \* If build is stable, Build Er. will only install it to production server also.

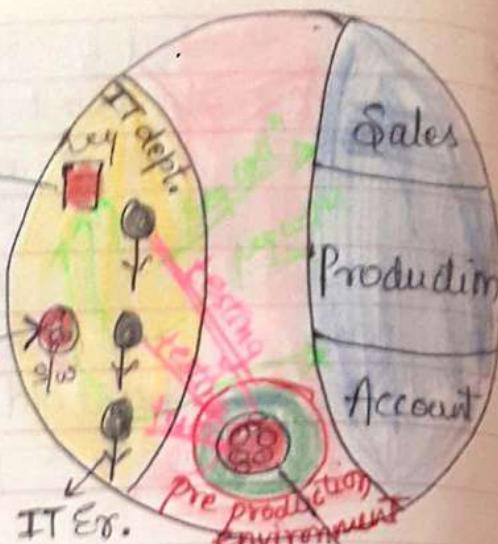
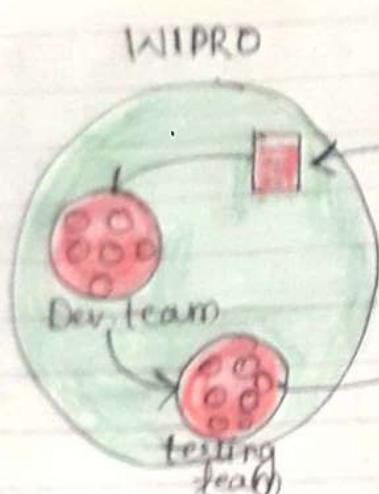
## Acceptance Testing



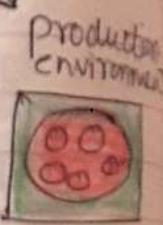
- \* time decided for the project was 31<sup>st</sup> dec, but after testing Test lead found at the end that sw is running well if we see externally but internally it is left up with some 60 critical defect & to fix them it would take some 15 days. He informed to the Proj. manager & tak tried to convince him but he told no, you just launch it otherwise we will have to pay them.
- \* so the sw was made release even with those 60 critical defects. Now at customer's place it they got the sw & where trying to install production server which took them nearly 15 days. Now they have installed & where running the sw & they found a defect & then informed to the wipro.
- \* By that time these bugger have also fixed all 60 critical defect & so they told ok we reatify your problem & its just the small issue & then they install the new build in which all the 60 se defects were fixed & now everything was running beautiful & even the fedax was also & also not made aware about 60 defects this way was the trick played by project manager & they escaped.
- \* But there are the companies which which might be older than wipro & much aware about sw & even have their own IT depts. & their IT engg. come in role, in those sort of situations these wipro people can't cheat them.

took 15 days.

PEPSI



- \* It engineers collect the requirement from different depts. & pass it to many s/w companies (diff. modules to diff companies) & once the agreement is signed & s/w is developed & delivered to these company they will first test it in pre-production environment & if working fine then only they will install it to production environment.



Why to IT branch?

All the s/w companies have sent the request to technical board that we want professionals who have technical knowledge of collecting the business requirements so conveying it to s/w companies & get their s/w prepared & can also understand & test the s/w & know all about it. And then after the IT branch was introduced.

CS people are technically good they can develop whatever the req is given to them but they don't know about the actual scenarios of how the particular business runs.

- \* IT engineers do acceptance testing, they can do all FT, Integration testing, sys testing but generally they do end-to-end testing.
- \* Is there any difference b/w end-to-end testing done by test engineer & end-to-end testing done by IT engineer  
→ yes, TE will do according to CRS, IT people will do it according to practical need of the business & customers.
- \* In 1st phase end-to-end testing done by TE & IT er. will be 80% same & 20% (diff. done by IT people) this gap increases as the no. of years with that s/w red.
- \* But TE's may leave the company if they get good experience - so IT companies are paying high for documenting things, so that it can be referred later also in case of long term projects.

Q. What is acceptance Testing?

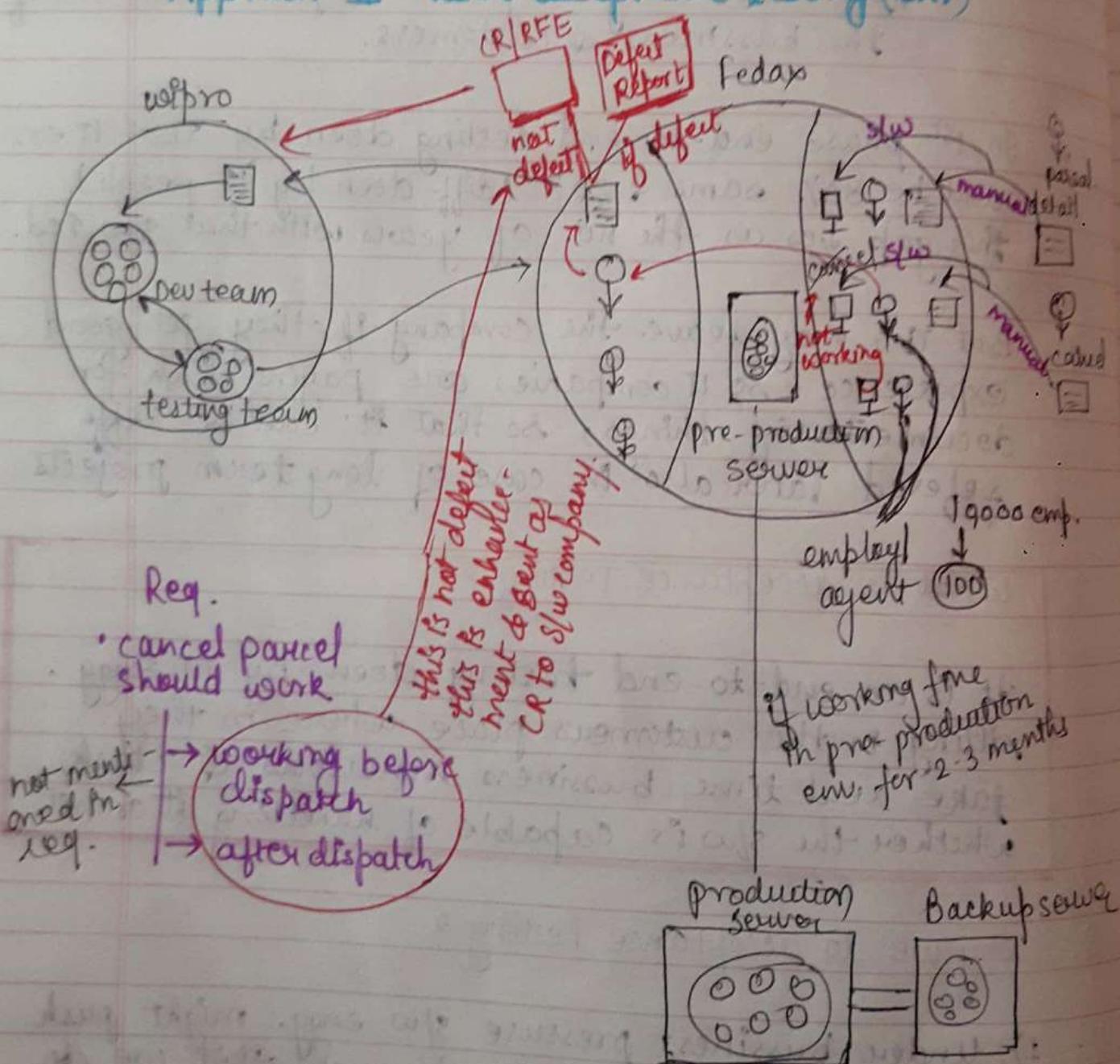
It is an end-to-end testing done by IT Engg. sitting in the customer's place where they take real time business scenarios & check whether the s/w is capable of handling it or not.

Q. Why we do acceptance testing?

- 1). Under business pressure s/w engg. might push the s/w with lots of bug, to avoid that we do acceptance testing.

- 2). With critical bugs if customer move the site to production, he will be under severe loss, to identify that we do AT.
- 3). Developers might misunderstand the requirements & developed wrong features, to identify that they do acceptance testing.

## Approach-II User acceptance testing (UAT)



\* CR (change request)

RFE (Req. for enhancement)

- Q. If IT Er. approves the app then which report they sent back to s/w company?

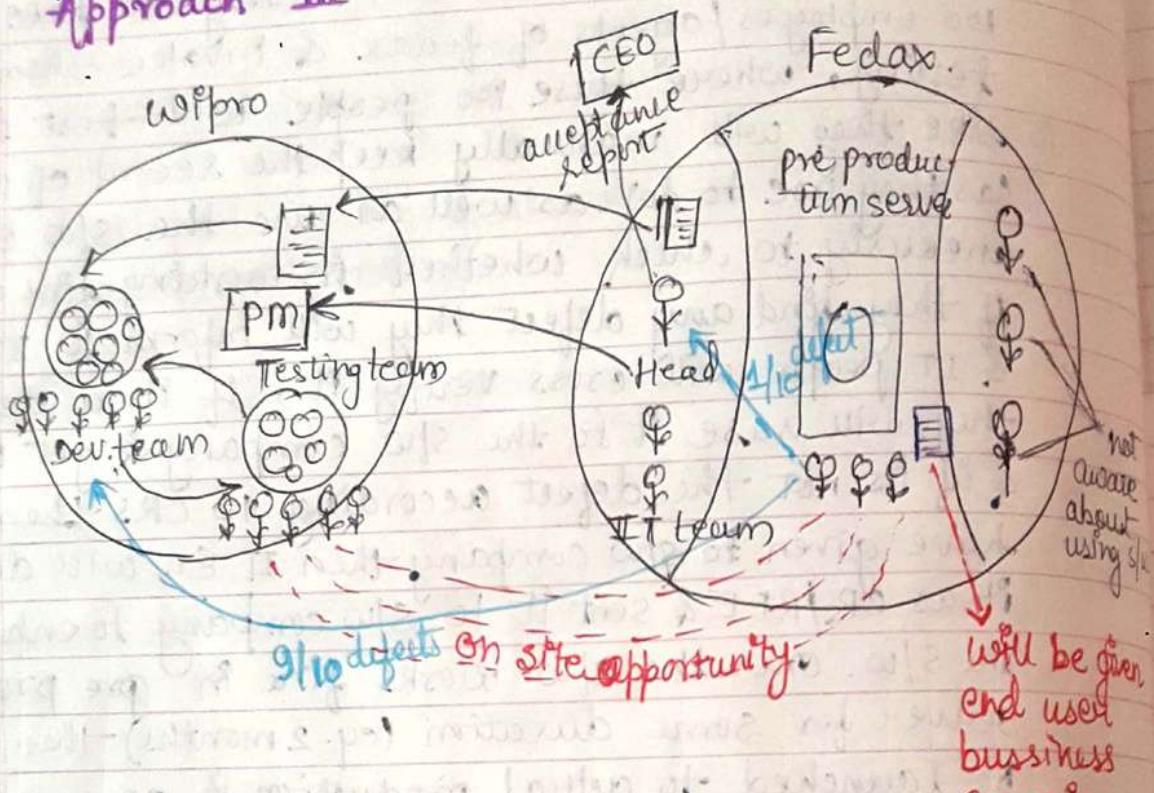
As IT team of Fedax is small, so they will take some 100 employees/agents of fedax & involve them for testing, where these 100 people will test the s/w like they will manually keep the record of data (as they use to do) as well as use the s/w simultaneously to check whether its working fine or not, if they find any defect they will inform to IT Er. & IT people will cross verify it, If its a defect they will raise it to the s/w company & get it fixed & if its not the defect according to CRs then they have given to s/w company then IT Er. will document it as CR/RFE & sent it to s/w company to enhance it the s/w. Once the s/w works fine in pre production server for some duration (e.g. 2 months) then it will be launched to actual production & now no manual work will be done.

Definition: It is an end-to-end testing done by end users when they use the s/w for the business for a particular period of time & check whether its capable of handling real time business scenarios.

- Q. Why the no. of acceptance test cycle 1 is?
  - ① After delivering the product, customer is getting lot of new ideas so he is asking for lot of changes.
  - ② The requirement which was given in the beginning is not clear so it is leading lot of rework.
  - ③ Testing team has not done proper testing & they have missed lot of bugs.

Q. If change in req. happens in approach-III & IV then this report is called what?

### Approach - III



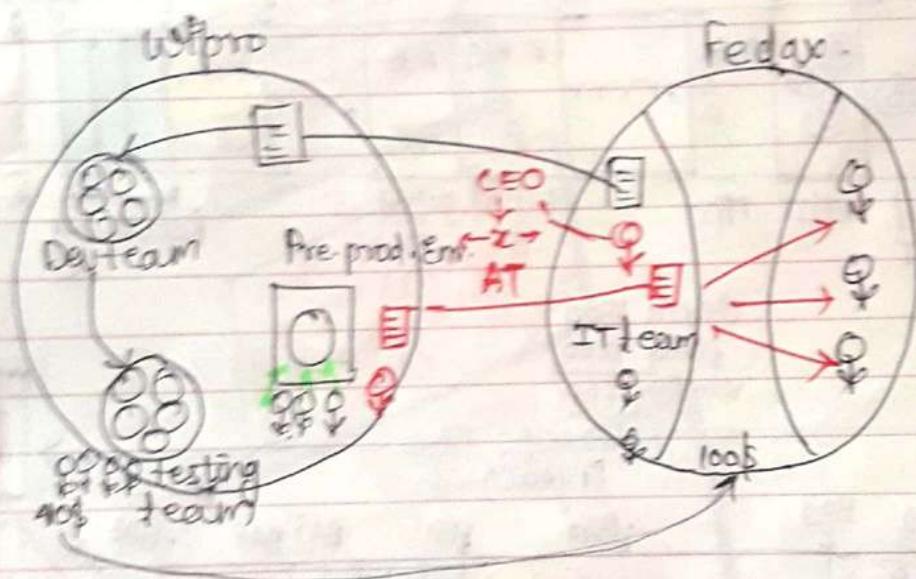
- \* IT team is small & end users/employees are not much aware about the s/w usage, so IT head will ask the project manager to send some of these tester in there place to perform acceptance testing. Now this is called onsite opportunity for TE's and IT team will collect all the requirement & generate end user business scenarios & ask these TE's to do acceptance testing referring to it.
- \* Now TE's at customers place, if finds any defect will inform to IT team, but internally a game is happening that out of 10 defects 1 will be told to IT head & rest 9 will be directly told to development team & will be fixed in the name of that 1 defect.
- \* This story is known to IT head also, but he is

more focused on quality of sw & other than making money from bugs.

Definition:

It is an end-to-end testing which is done by our own test engineer sitting at the customer's place, where ~~or~~ in they refer end user business scenarios & they check whether B/U is capable to handle it.

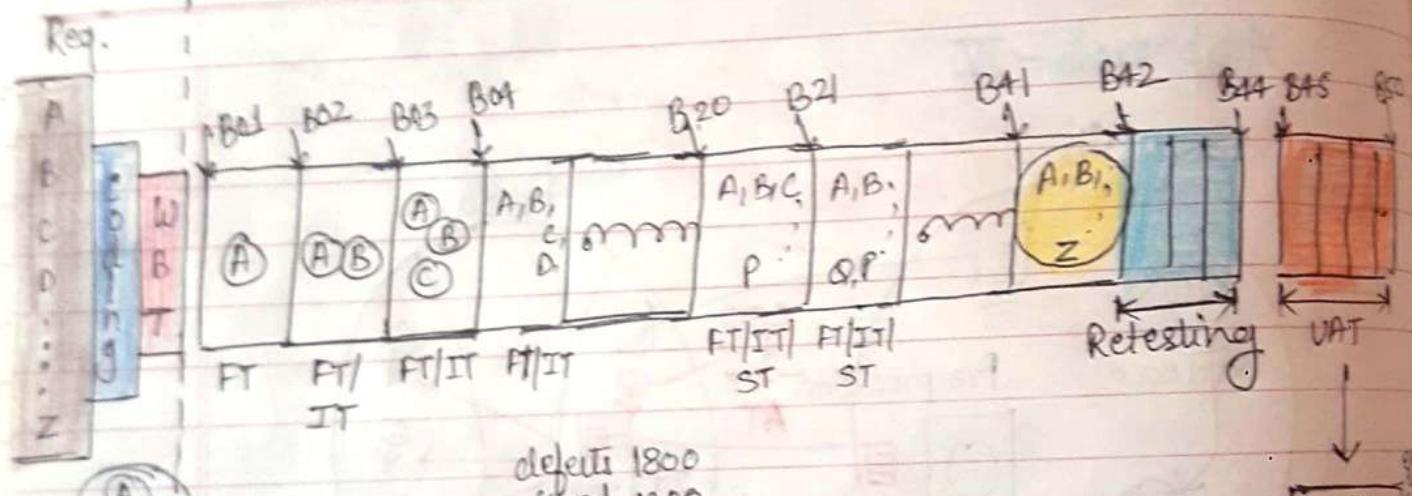
## Approach - IV



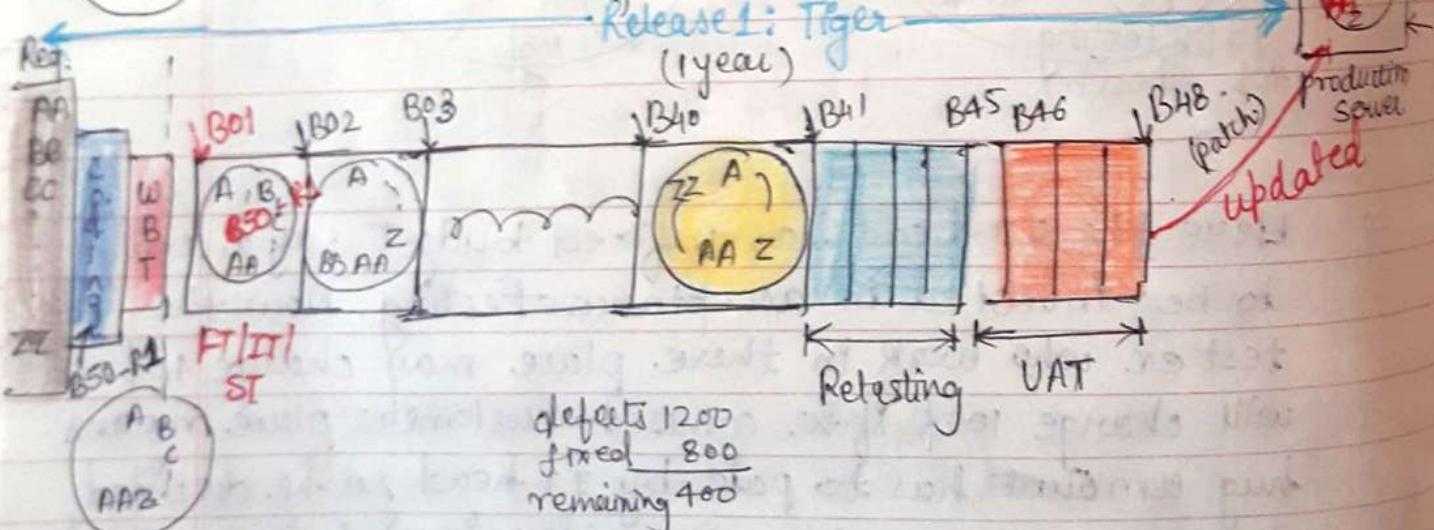
- \* Here the IT head has a fixed budget say x amt to be invested in acceptance testing. Now the tester who work in there place may charge 40\$ will charge 100\$. If he comes to customers place means huge amount has to paid by IT head so he decided to collect the req (business scenarios) & documented them & went to the customers place. Now here he takes some TE's & pay ask them to perform AT for those sw. by reviewing that all those business scenarios.

**Definition:**

It is an end-to-end testing done by our own TEs, sitting in their own place. They are given the business scenarios & they test whether the shop is capable to handle of handling it or not.



Release 1: Tiger  
(1 year)



Release 2: Lion  
(1 year)

## Release 3



will stop developing  
BBB

critical defect found in  
S/W, which is already  
installed in production  
server. (by customer)

↓ call  
Project manager  
↓ call urgently  
Dev. & testing  
team

↓ will take B48  
of 2nd Release &  
work day & night  
to fix that defect

↓ meeting by PM

} Hot fix

: Bug found at customer  
site & it needs an  
immediate fix

if customers ask  
to develop for  
new req. in  
B/W

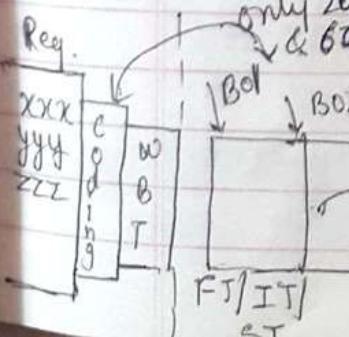
reasons of  
that defect

defect

RCA (Root cause analysis)  
JIGSAW (incident management)  
Ishikawa technique  
FBT (fishbone technique)

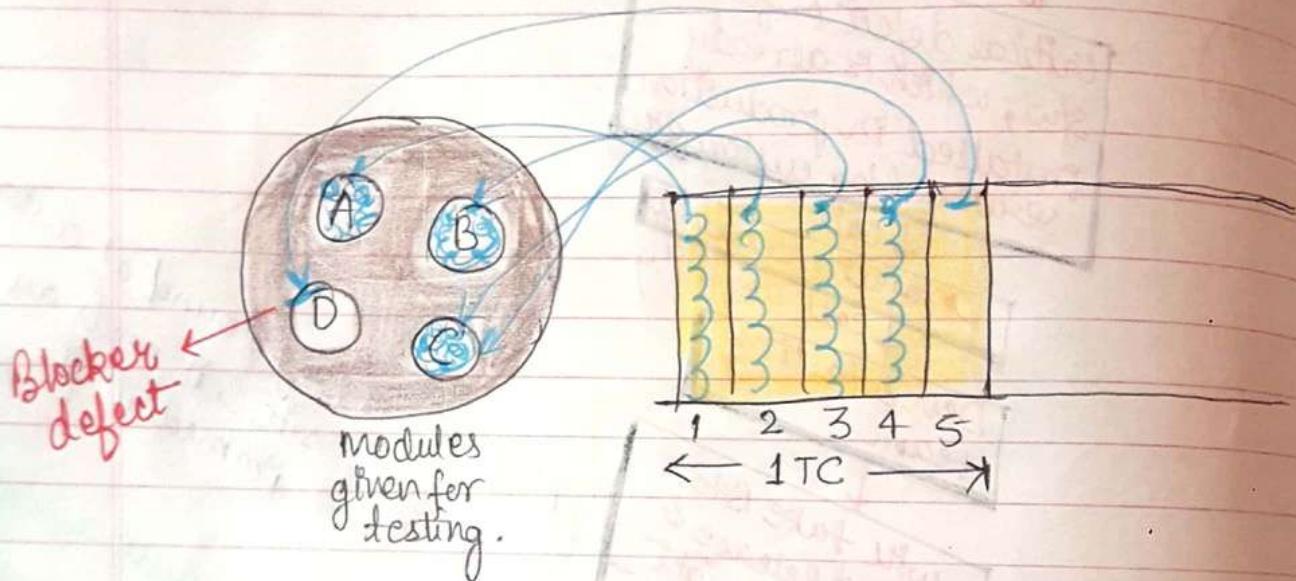
Release 3: short term →

release  
(15d, 20d, 30d, 6m)



Retesting \* UAT

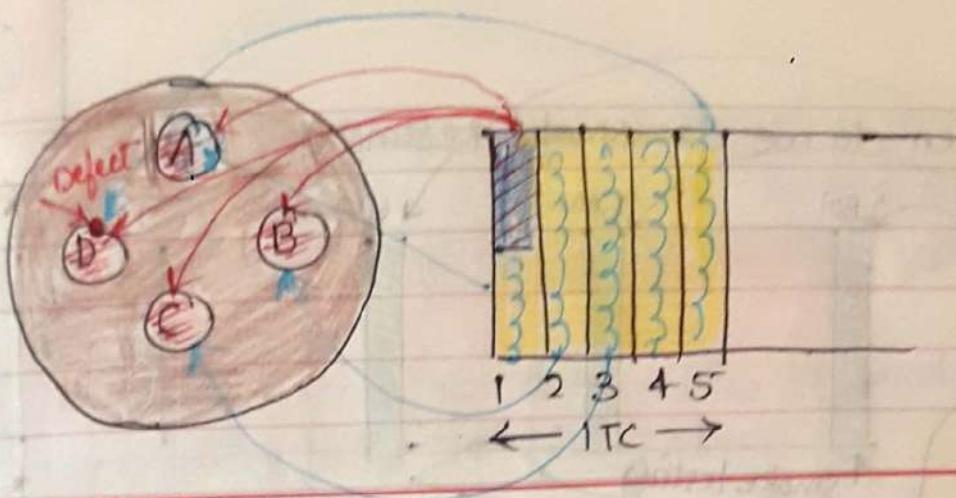
# Smoke testing / Sanity testing / Dry-run testing / Build verification testing



1<sup>st</sup> day he tested module A thoroughly, 2<sup>nd</sup> day he tested module B thoroughly, 3<sup>rd</sup> day he tested half of module C, 4<sup>th</sup> day he tested remaining half of module C. 5<sup>th</sup> day he tried testing module D & he faced blocker defect. He reported to development team they asked for 3 days to fix it & so the 1<sup>st</sup> cycle will be delayed by 3 days & similarly next test cycle will be delayed by 3 days & there by the 1<sup>st</sup> release will be delayed by 3 days. Who's mistake is it?

TE ✓

Dev. team



### Definition.

Testing the basic / critical features of an application before doing thorough testing is called smoke testing.

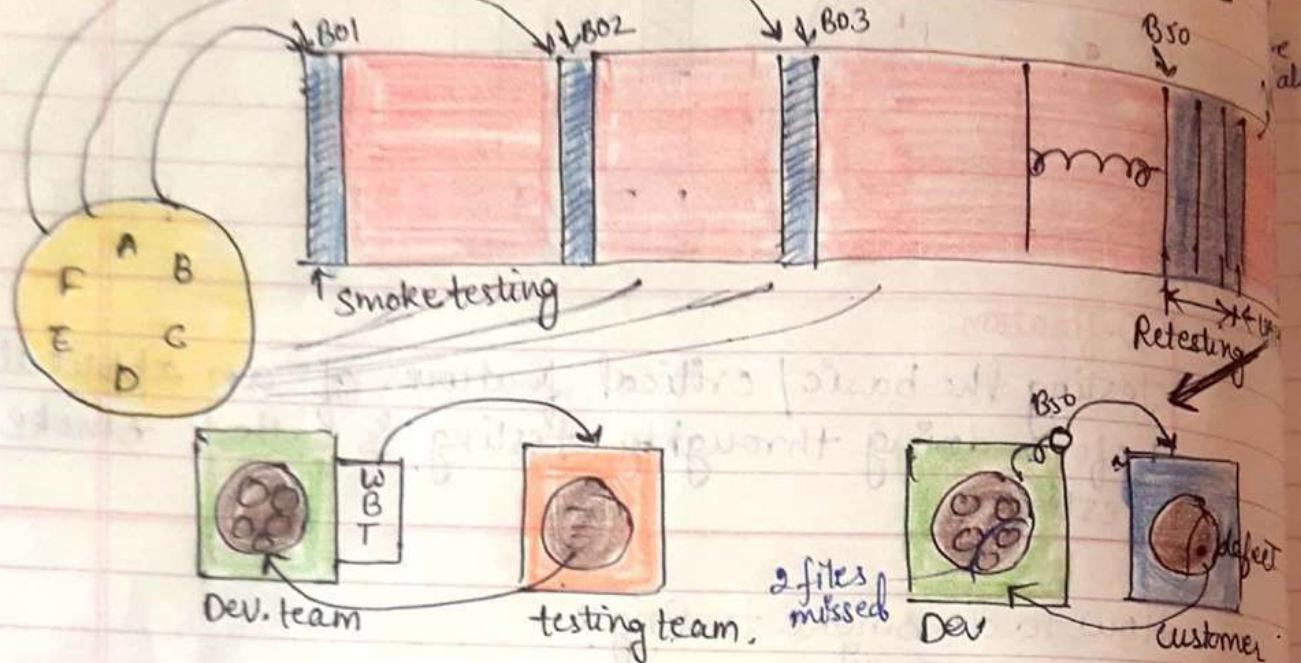
How to do smoke testing?

gmail	
features to be tested as smoke testing	features not to be tested in smoke testing.
signup	Sent Item
login	forgot pw
compose	logout
Inbox	Labels
Sent Item	Help.
Draft	
All mails	

### Assignment 7

For FB & whatsapp find the features to be tested in smoke testing & not to be tested in smoke testing.

Q. When do we do smoke testing?



When ever the development team gives build to the testing team, we will always start with smoke testing.

When ever the development team gives the build definitely they would have done some changes & those changes may affect basic or critical features so we start with smoke testing.

While doing UAT at customer side also, 1st smoke testing should be done because if they get defect later & get new build (after fixing the defect) they need to test it again from the begining & that will be the waste of time so smoke testing should be done at the beginning itself to know about blocker defects.

When ever the build goes for acceptance testing, customer should always start with smoke testing.

In order to ensure that installation is done properly we do a round of smoke testing after s/w is installed on production server.

Developer soon after doing WBT, do smoke testing before giving build to testing team.

Q. Why we do smoke testing?

① To check whether the build is testable or not.

② Testing the basic feature in the beginning itself & if we find any defects sent it to developer in the beginning only so that development team will have sufficient time to fix the defect.

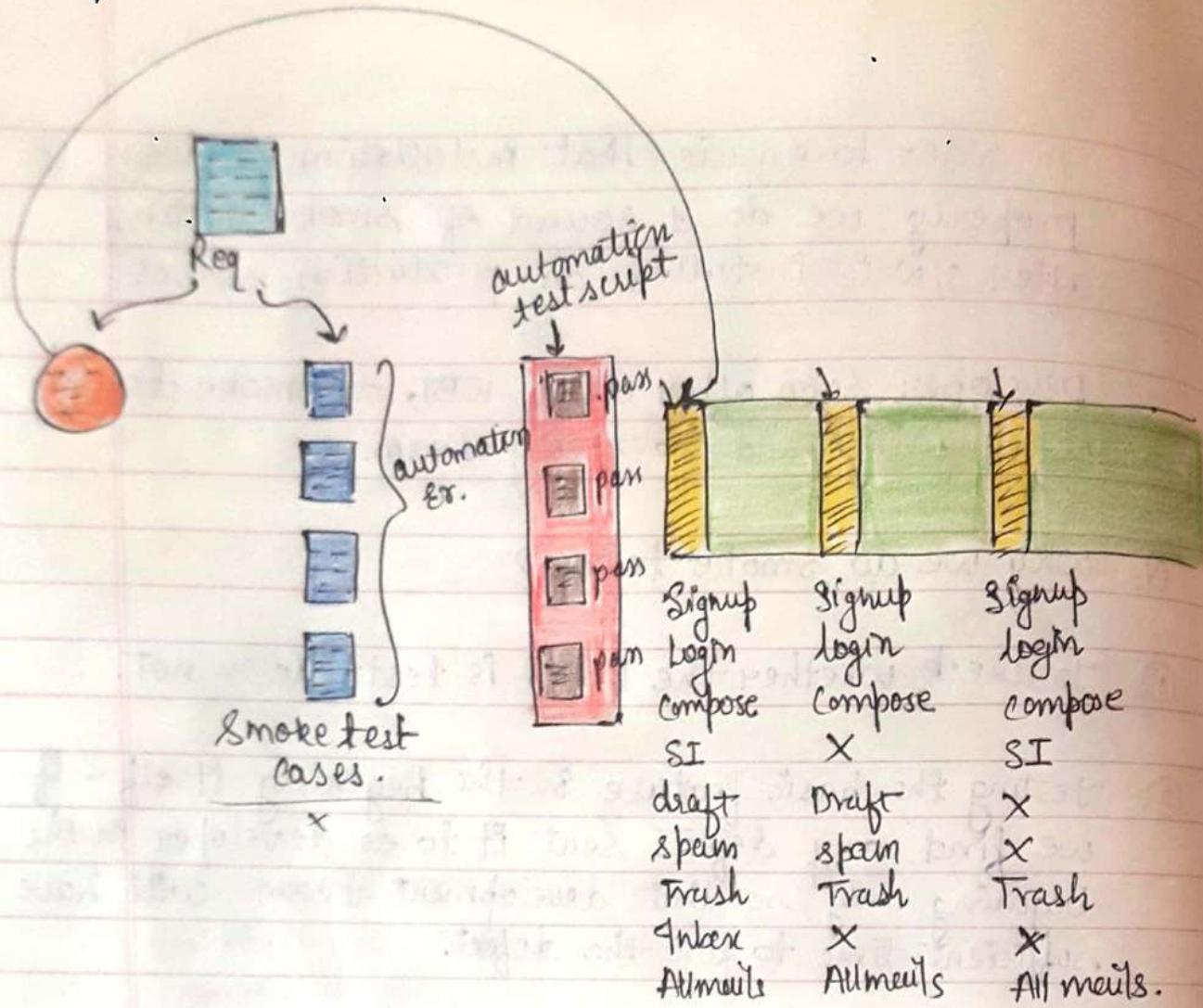
③ Doing smoke testing indirectly confirms that the product is installed properly.

④ It is a kind of health check of a s/w where in we try to check, whether we received broken build from development team. It is also called as Build verification testing.

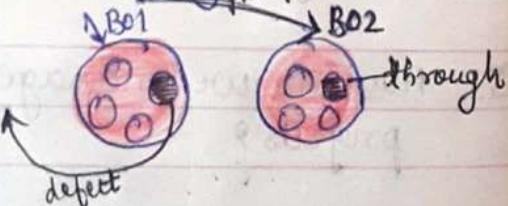
Q. How do we manage smoke testing in real time projects?

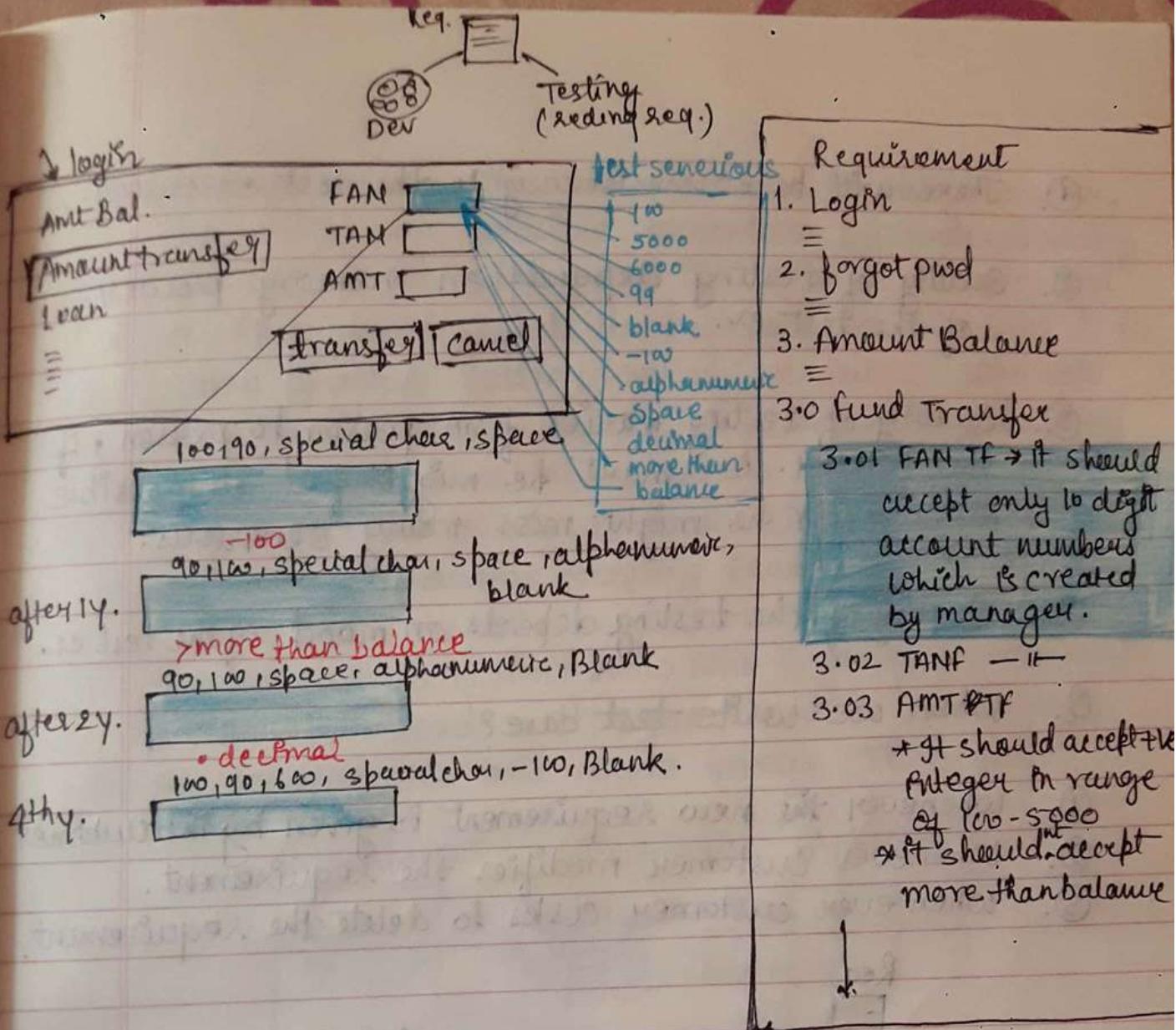
- By executing smoke test cases  
(or)

By executing automation test script.



Difference between smoke testing & sanity testing.

smoke testing	sanity testing.
<ul style="list-style-type: none"> <li>→ wide &amp; shallow testing.</li> </ul> 	<ul style="list-style-type: none"> <li>→ narrow &amp; deep testing</li> </ul> 
<ul style="list-style-type: none"> <li>→ It is scripted</li> <li>→ done by dev. &amp; test ex.</li> </ul>	<ul style="list-style-type: none"> <li>→ It is unscripted</li> <li>→ done by test ex.</li> </ul>



### Q. What is test case?

It is a document which consists of all possible scenarios for a specific requirement.

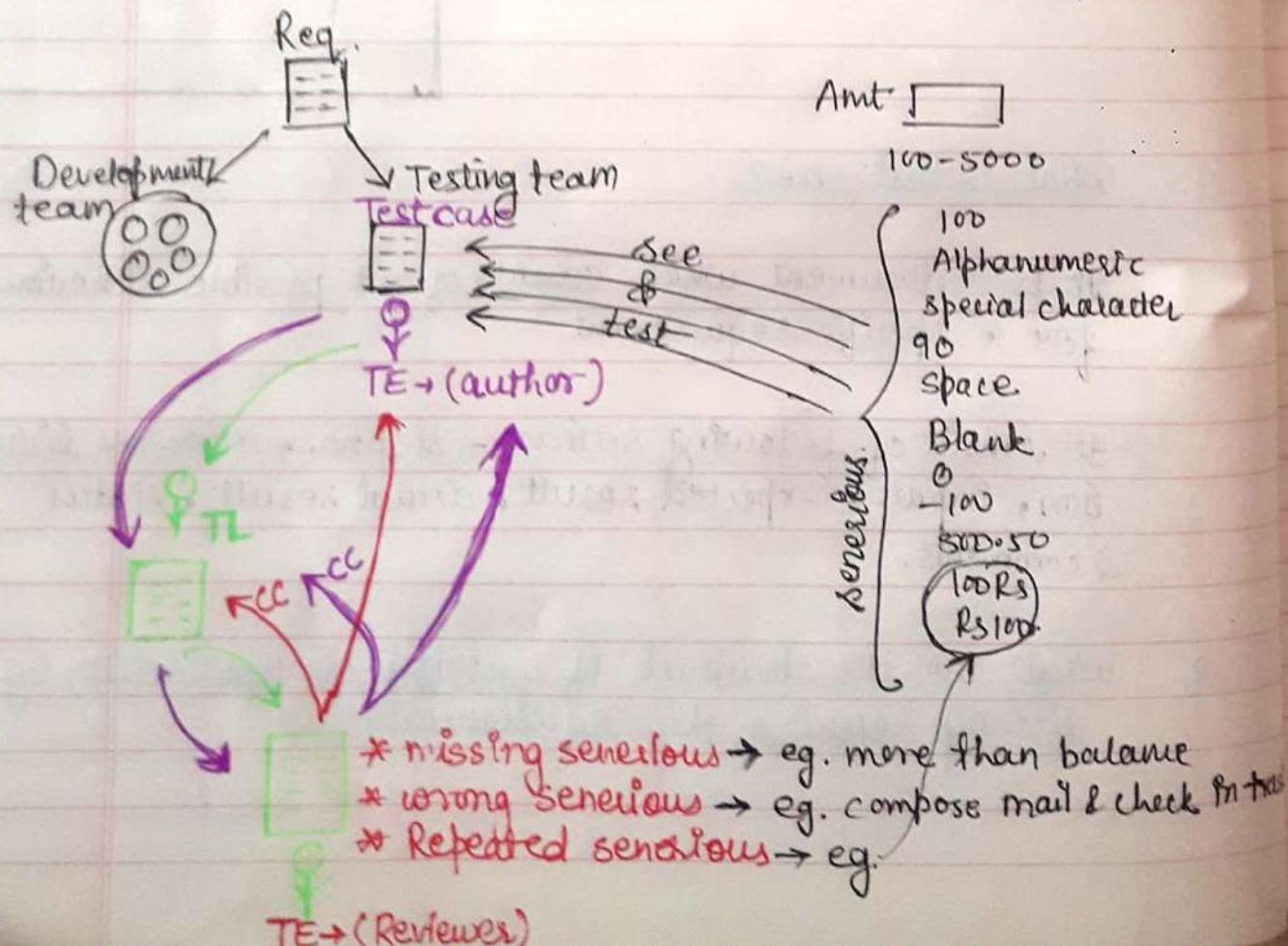
It consists of following sections - step no., action or description, inputs, expected result, actual result, status & comments.

### Q. what are the drawbacks if you test the application by directly referring the requirement.

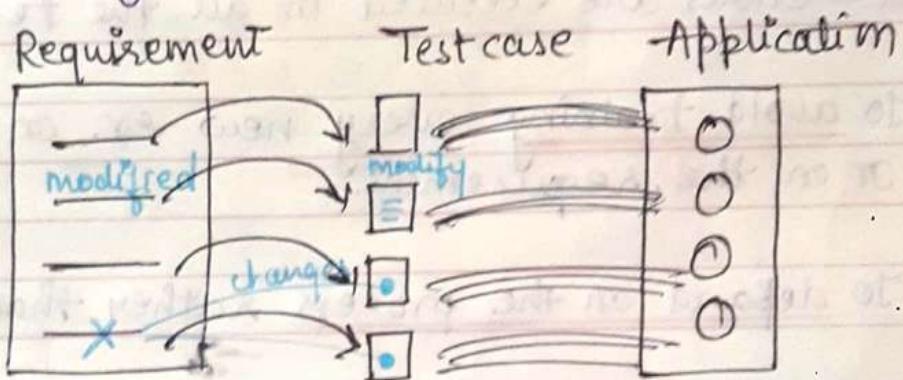
- ① There will be no consistency in the test execution.
- ② Quality of testing depends on memory power of the tester.
- ③ Quality of testing varies from person to person, if the tester is smart he might find all possible scenarios or he might miss many scenarios.
- ④ Quality of the testing depends on mood of the tester.

Q. When we write test case?

- ① whenever the new requirement is given by the customer.
- ② whenever customer modifies the requirement.
- ③ whenever customer asks to delete the requirement.



Test er. who writes the scenarios is called **author**, he writes all possible scenarios by understanding the SRS & will document it which is then called as test case & sent it to test lead, he will sent it to some another test er. called **reviewer** who will review the test cases, while reviewing he may find **missing scenarios**, **wrong scenarios** or **repeated scenarios**. Reviewer will sent these mistakes to author keeping team lead in CC. Now author will correct the document & sent it back to team lead as he don't know who is going to review it & then team lead will again sent to reviewer. If no. of cycles are more mean TE is poor.



Testing the application by referring the **test cases** is called **slw testing**. If there will be modification in requirement, test case will also be modified & application will be tested according to modified test case. If some req. has been deleted by the customer, we will check if that deleted req. have affected to some other test cases, if so we will modify it & will re-check the application according to modified **slw test case**.

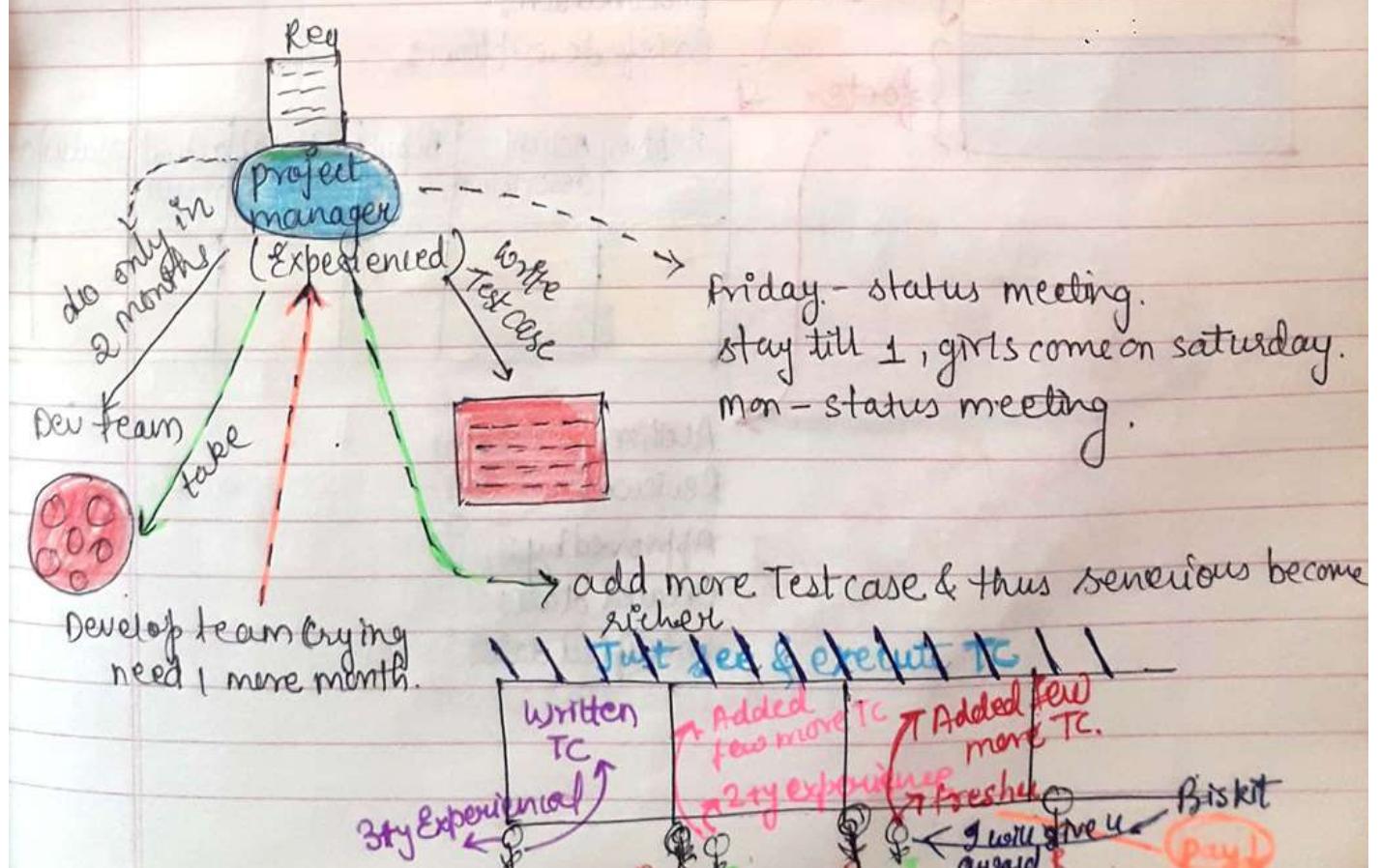
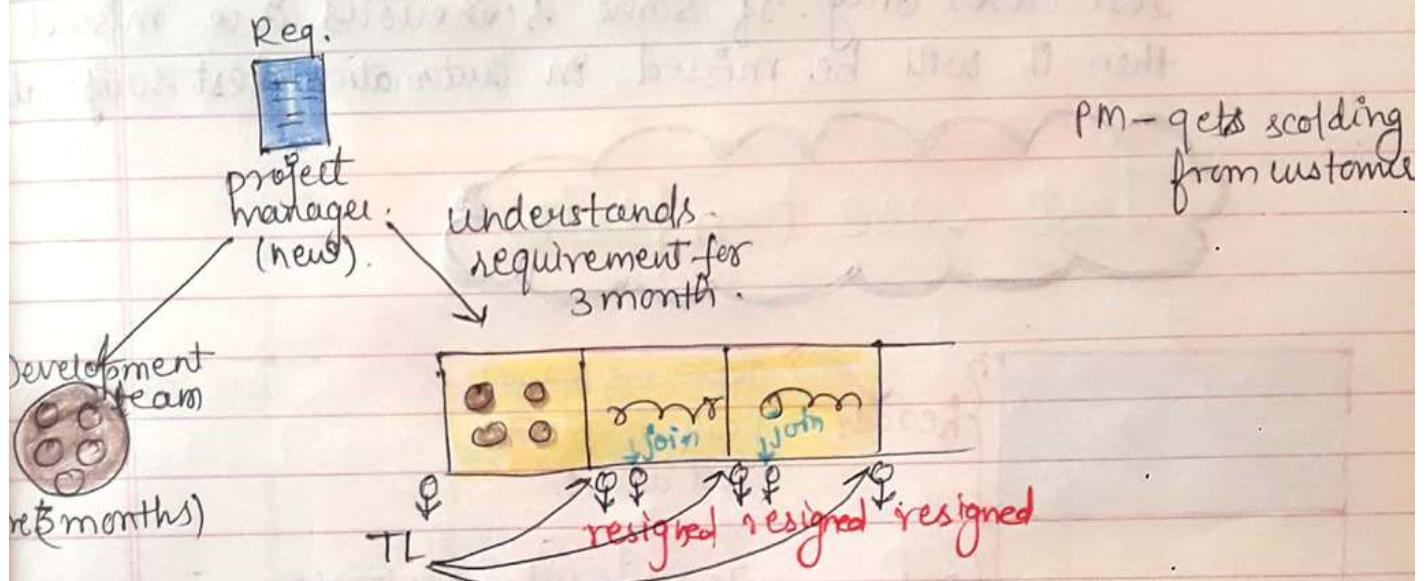
Q. Why we write Test case ?

- ① To have a better coverage -  
when developer will be developing the product, the testing team will have a sufficient time, test er. should identify all possible scenarios & document it so that when ever the development team gives a built for testing, testing team will not miss any scenarios while test execution.
- ② To have a consistency in test execution - means if you would have documented all possible scenarios you can ensure that all the scenarios are executed in all the test cycle.
- ③ To avoid training every new er. on the product or on the requirement.
- ④ To depend on the process rather than a person.
- ⑤ Test case is the only document which acts like a proof to developer, customer, project manager & management that they have covered all possible scenarios.
- ⑥ Test case, it acts like a base document to write automation scripts - if you write automation script by referring Test cases you can ensure that same coverage is maintained in automation script as well.

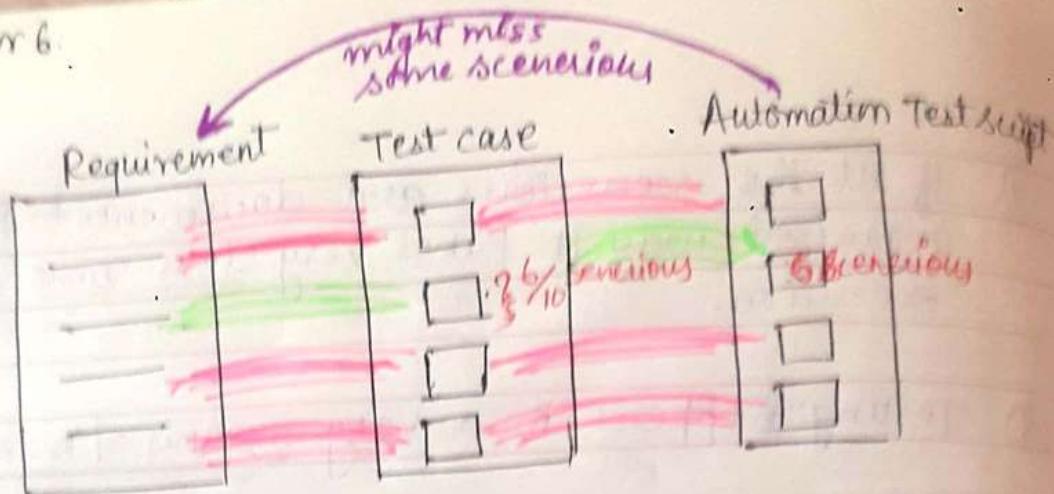
⑦. If all the scenarios are documented in very organised way it takes very less time for test execution.

⑧. Testing happens in a very organised manner.

Explanation for 3 & 4.

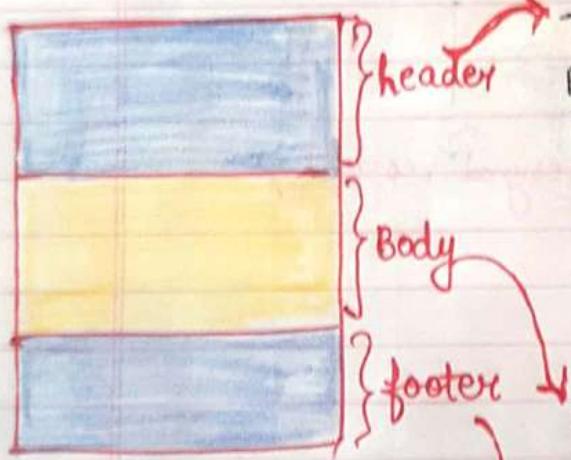


Explanation for 6



test script will be written by referring manual test cases only. If some scenarios are missed then it will be missed in automation test script also

## Test Case template



Test case Name:  
Requirement no:  
Test data:  
Severity:  
Test case type: FTC / ITC / STC  
Precondition:  
Brief description:

Step No.	Action description	Input	expected result	actual result	Status	Comments

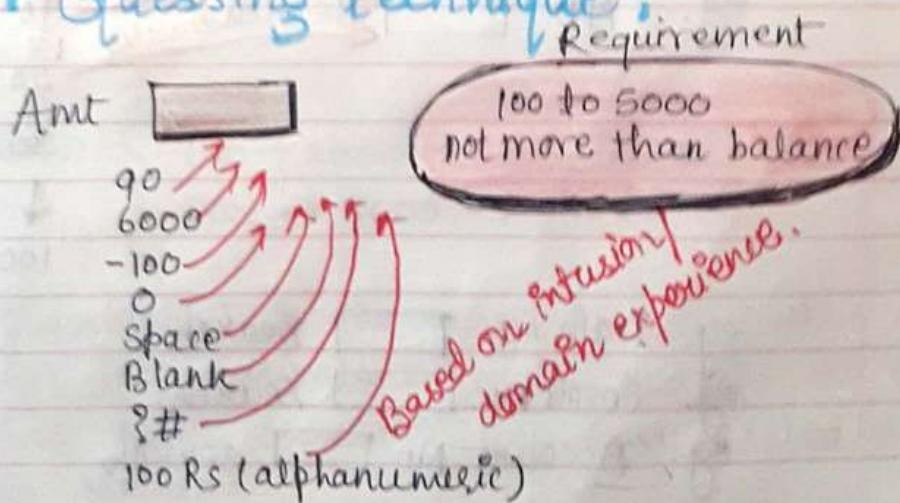
Author:  
Reviewer:  
Approved by:  
Created date:  
modified date:  
/ / / / / / /

If we randomly derive the scenarios for a specific requirement we can't ensure that we have covered maximum possible scenarios to over come this we should follow testcase design technique while deriving the scenarios.

## Test case design technique

- Error guessing technique
- Equivalence partitioning
  - Pressmen
  - Practice
- Boundary value Analysis

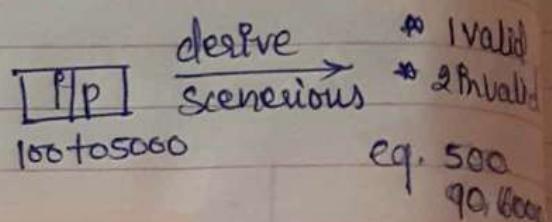
### Error Guessing technique :



Definition: When we guess all possible errors based on intuition or based on passed experience it's error guessing technique.

## Equivalence Partitioning technique:

- pressmen
- Practice



### Pressmen technique:

According to him:

- ① If the given IP is the range of value then derive the scenarios for 1 valid & 2 invalid values.

e.g. age  
temp  
marks.

- ② If the given IP is the set of values then derive the scenarios for 1 valid & 2 invalid values.

e.g. PID  Search

~~invalid~~ ~~valid~~ 9

10 - mobile

15 - scanner

30 - printer



1000 products

e.g. train no.  search

e.g. Enroll No.  search

e.g. Account No.  search

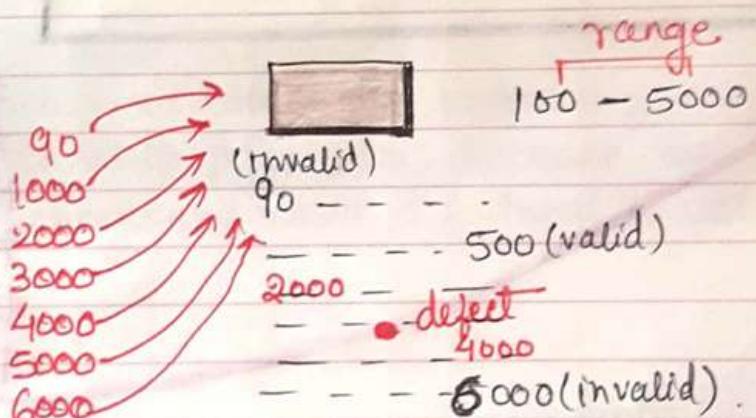
③ If the I/p is boolean then derive the scenarios for both true & false value.

e.g. gender       male  
                      female

select - True  
unselect & click on submit - False.

e.g.  I Agree.  
select - True  
unselect - False

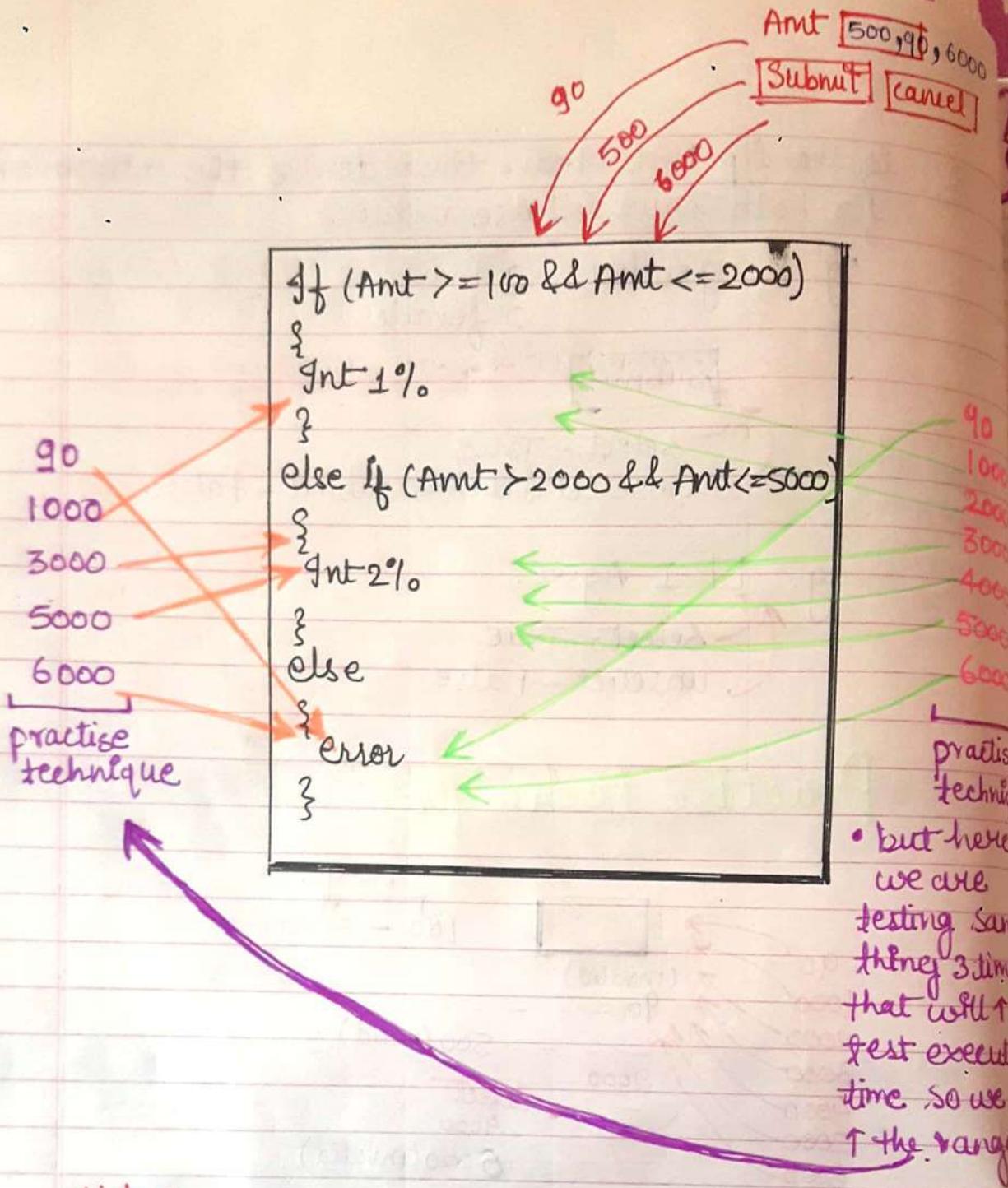
## Practise technique



Deviation in value

$100 \rightarrow 2000 \rightarrow 1\% \text{ interest}$
$2001 \rightarrow 5000 \rightarrow 2\% \text{ interest}$

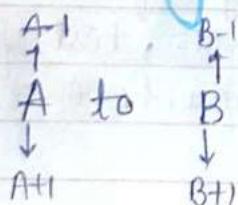
Definition: if the given Input is the range of value then divide the range into equivalent parts, test for all the values but make sure that you have tested at least 2 invalid values.



Note :

When ever there is deviation in the requirement go for practise.

## Boundary Value Analysis (BVA) technique

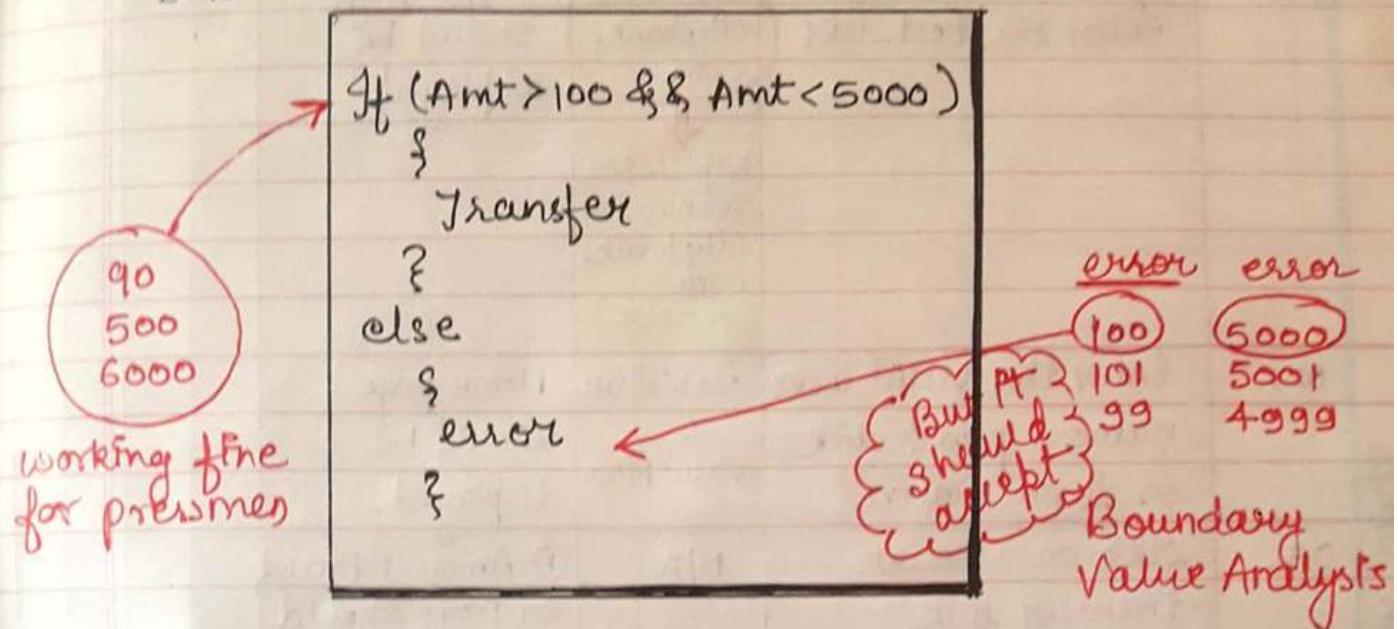


e.g.

100 - 5000  
99 4999  
100 5000  
101 5001

testing by using pressmen technique:  
we havn't tested elseif part of  
the program.

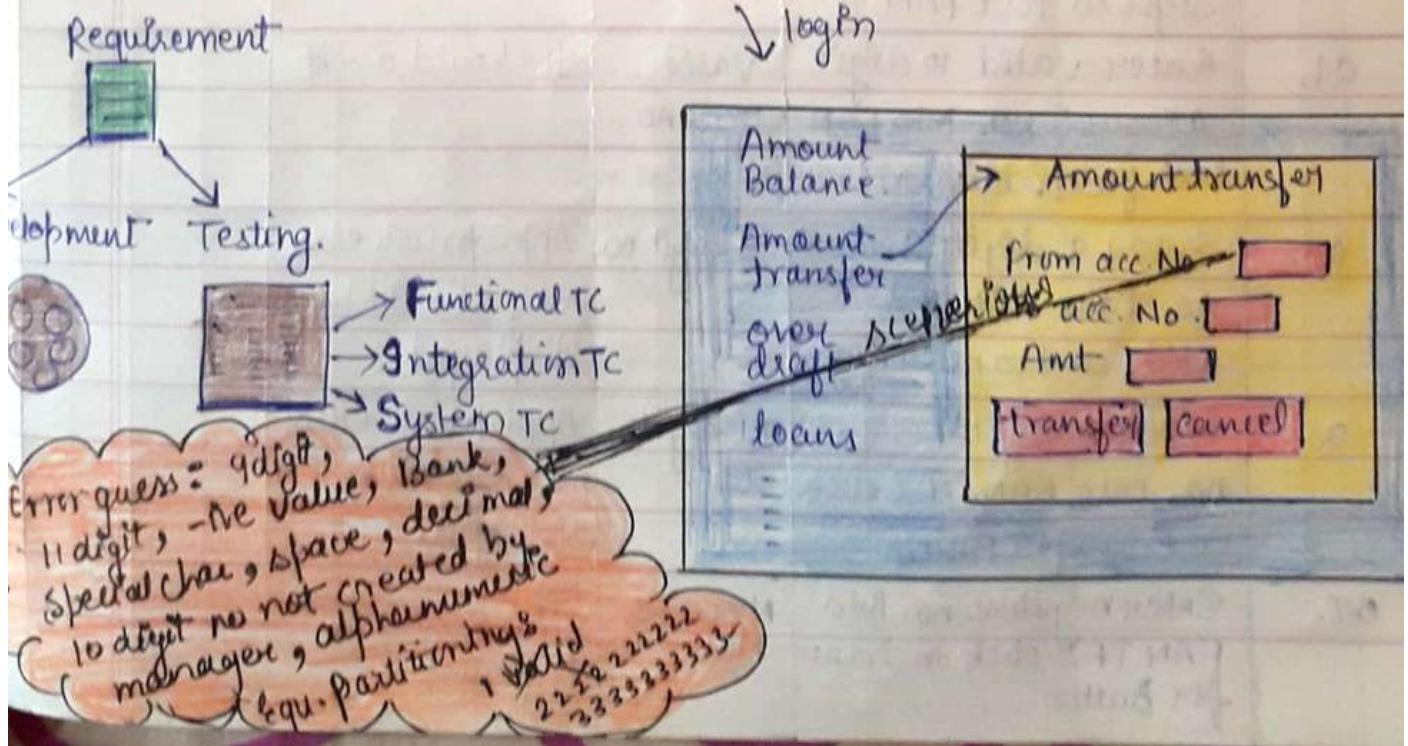
Definition: If the SPP is range of values between A to B, then derive the scenarios for A, A+1, A-1 & B, B+1, B-1.



## Notes

when ever we go for boundary value analysis we can skip pressmen, because BA BVA itself have checked 1 valid & 2 invalid values.

## Functional test case:



eg: functional test for amount transfer feature.

QA	track	Test
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Test Session

Step No.	Action / Description	Inputs	Expected Result	Actual Result
01.	Open the browser Enter the test-URL	Hardcoded  http://QA city bank. com	Login page should be displayed.	
		http://<Session ver name> city bank. com (generic)		
02.	Enter the valid user name & p/w & click on login button.	Valid UN & Valid P/w	Home page should be displayed.	
03.	Click on amount transfer link	N/A	① Amount transfer page should be displayed. ② The following components should be displayed. * FAN TF * TAN TF * Amt TF * transfer button * cancel button.	
<b>Navigation steps</b> 				
04.	following are the steps to test FAN TF			
04.	Enter valid 10 digit account no. into FAN text field & click on transfer	valid	It should accept	
05.	Enter 9 digit account no. into FAN TF & click on transfer Button	9 digit no.	Appropriate error msg should be displayed	
06.	Enter 11 digit account no. into FAN TF & click on transfer Button	11 digit no.	Appropriate error msg should be displayed.	
07.	Enter negative no. into FAN TF & click on trans- fer Button	Negative No.	Appropriate error msg should be displayed.	

Step No.	Action / Description	Inputs	Expected Result	Actual status	Comments
08.	without entering any value into FAN TF click on transfer button.	Blank	Appropriate error msg should be displayed.		
09.	Enter special character into FAN TF & click on transfer Button.	special character	Appropriate error msg should be displayed.		
10.	Enter the no. space in b/w the no. into FAN TF & click on transfer Button.	space in between no.	Appropriate error msg should be displayed.		
11.	Enter alphanumeric no. into FAN TF & click on transfer button.	alpha numeric no.	Appropriate error msg should be displayed.		
12.	Enter decimal no. into FAN TF & click on transfer button.	decimal no.	Appropriate error msg should be displayed.		
13.	Enter 10 digit no but it is not created by manager.	invalid acc. no.	Appropriate error msg should be displayed.		
14-26	following are the steps to test TANTF				
27.	same as for FANTF just change FAN → TAN				
27.	Enter the same account no. in TANTF as FANTF no	valid acc. no	Should not accept		
28.	following are the steps to test Amt TF				
28.	Enter an integer no. with in the range 100-5000 into amt TF	positive integer no. b/w 100-5000	Should accept		
29.	Enter a no. less than 100 than 5000 in amt TF	→ 100 → 5000	Should accept		
30.	Enter 5000 in amt TF	5000	Should accept		
31.	Enter no. less than 100 in amt TF	→ 100 99	Should not accept & appropriate error msg should be displayed		
32.	Enter no. greater than 5000 into amt TF	→ 5000 5001	Should not accept & appropriate error msg should be displayed		
33.	Enter any negative no. into Amt TF	negative no.	Should not accept & appropriate error msg should be displayed		

Amt No. → BVA → 99 ✓ 4999 ✓  
 100 ✗ 5000 ✗  
 101 ✓ 5001 ✓

Step No.	Action / Description	Inputs	Expected Result	Actual Result	Status
34.	Enter a decimal no. into amt No. text field	decimal no	Should not accept decimal & appropriate error msg should be displayed.		
35.	Enter a alphanumeric no. into amt No. textfield	alphanumeric no.	Should not accept & appropriate error msg should be displayed		
36.	Enter special character into amt No. text field	special character	Should not accept & appropriate error msg should be displayed.		
37.	Enter Don't enter anything in amt No. TF	blank	Should not accept & appropriate error msg should be displayed.		
38.	Enter Space in blno. In amt No. TF	Space in blno.	Should not accept & appropriate error msg should be displayed.		
39.	Enter 101 in amt No TF	101	It should accept		
40.	Enter 4999 in amt No. TF	4999	It should accept		
following are the steps to test transfer button & cancel button.					
41.	If all the fields - TAN, FAN & amt TF is filled with valid values, click on to transfer. (mandatory fields should be filled)	FAN, TAN, amt TF	Should navigate to next page conformation msg should be displayed.		
42.	If delayed in entering the clicking transfer button.	N/A	Time out msg should be displayed.		
43.	click on cancel button.		Reconfirmation msg should be displayed.		
44.	click on cancel button after reconfirmation.		Should navigate to next page.		

**Note:**

- \* while writing test cases we should always start with navigation steps.
- \* we should use words like "should be displayed" / "must be displayed."
- \* we should not hard code the test case we should write generic TC.

**eg: Integration test case for amount transfer between 2 account.**

Step No.	Action/Description	Inputs	Expected Result	Actual Result	Status	Comments
1.	Open the browser, enter the test URL	https://<server name>.city bank.com	Login page should be displayed.			
2.	Enter valid user name & password of user A & click on login button	Valid UN & PWD for A.	Home page should be displayed.			
3.	click on amt trans-fer link	N/A	Amt transfer page should be displayed.			
4.	Enter valid data into FAN, TAN & amt TF & click on		confirmation msg should be displayed.			
5.	click on amt bal-ance link		The transfer amt should be deducted.			
6.	Click on transaction link		The transaction details should be displayed.			
7.	Click on logout link		Login page should be displayed.			
8.	Enter Valid user name & pwd of user B & click on login Button	Valid UN & PWD for B	Home page should be displayed.			

9	click on Amt Balance link	The transfer amt should be added in B account.
10.	click on transaction link	The transaction details should be displayed.

Header :-

Test case Name : CityBank\_AmountTransfer\_CheckAB & transaction  
 (or) CityBank\_AmountTransfer\_IntegrationTC  
 (or) CityBank\_AmountTransfer\_all possible scenarios

Req. No: CBA-SRS 30.1.1

severity: critical / major / minor.

Test data: username, p/w & account No. of userA & userB.

Pre-condition: - main balance should be there in A's account  
 - user A's & B's acc. no. should be created

Testcase type: Integration TC (+ve & -ve)

Brief description: This test case consists of all possible integration scenarios on amt. transfer feature.

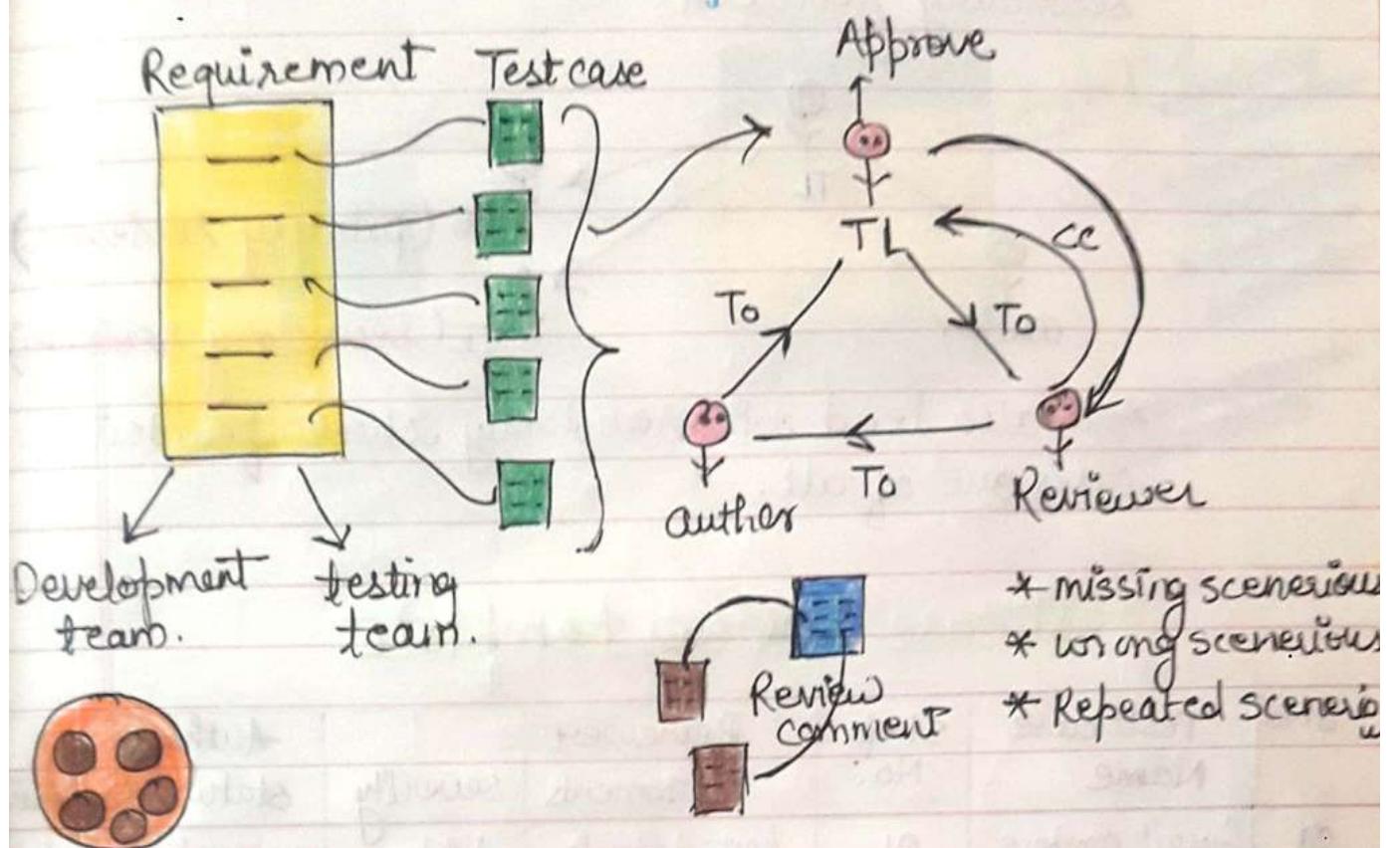
### Assignment-8

- 1). Write 5 test cases for gmail.
- 2). Write system test case on OD feature.

Best test case should have following

- ①. easy to understand
- ②. cover all possible scenarios
- ③. It should take very less time for execution.
- ④. Should not have repeated & wrong scenarios.

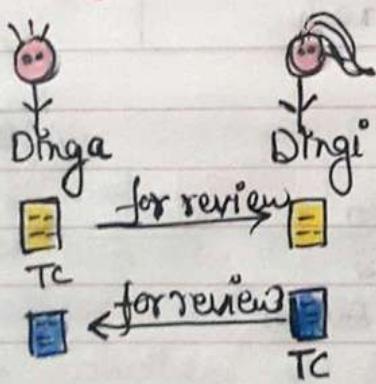
## Test case review process:



Q. How TL will allocate the test case for reviewing?

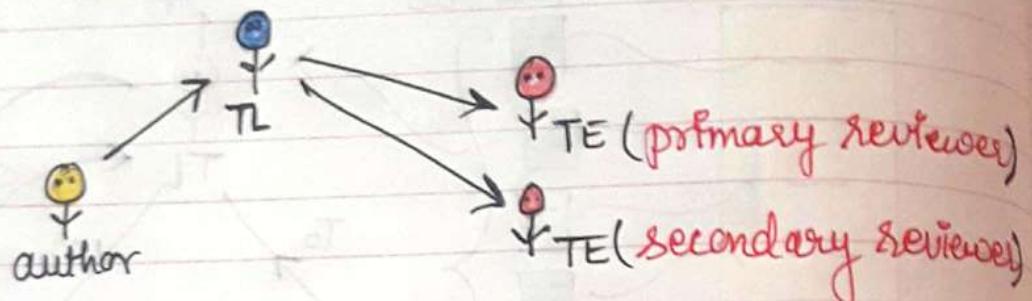
- Based on domain knowledge of TE
- one who works on similar module (home loan - personal loan)
- assign the test case to one who takes responsibility.

Q. How do you ensure that test er. will review test cases properly?



both r finds so they may not review

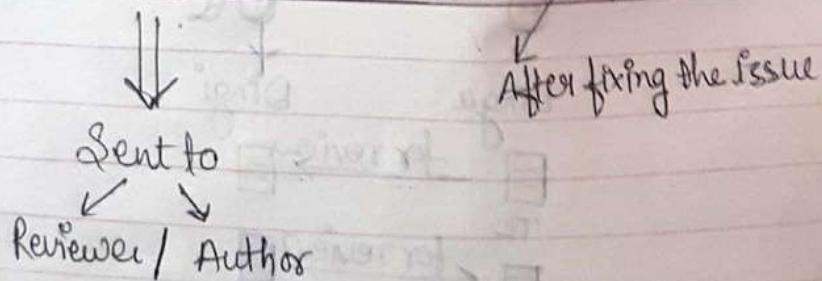
Ans. assign same testcase to both primary & secondary reviewer.



And also lead will randomly check few test cases out of all.

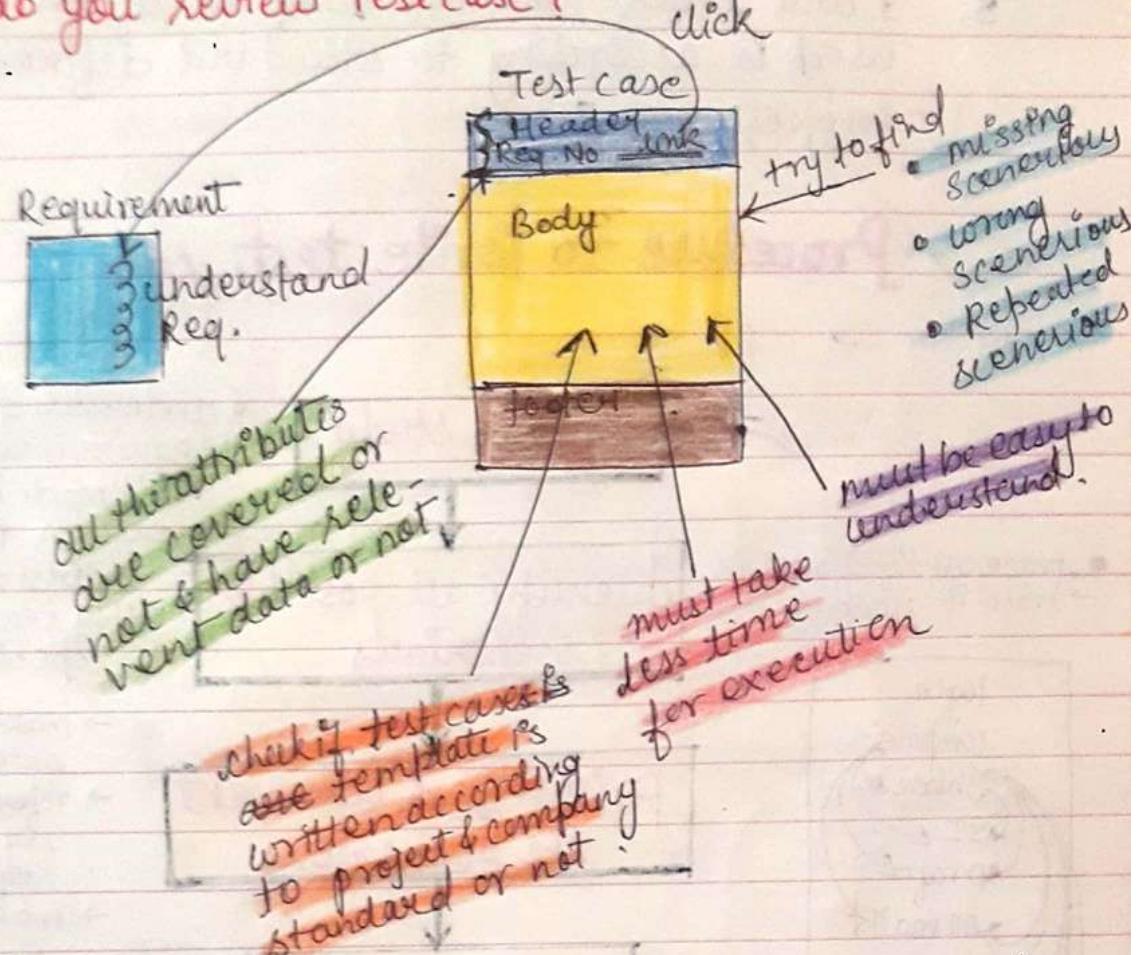
## Test case Review template

SNo.	Test case Name	Step No.	Reviewer comments	Severity	Authors status	comment
01	Gmail-compose	01	test data is missing	high	No fixed	Test data is not required
02	Gmail-compose	03	Checking sent mail in S.I scenarios is missed	critical	fixed	required
03	Gmail-compose	08	wrong scenarios checking sent mail in trash	high	fixed	



- Must be approved by TL.

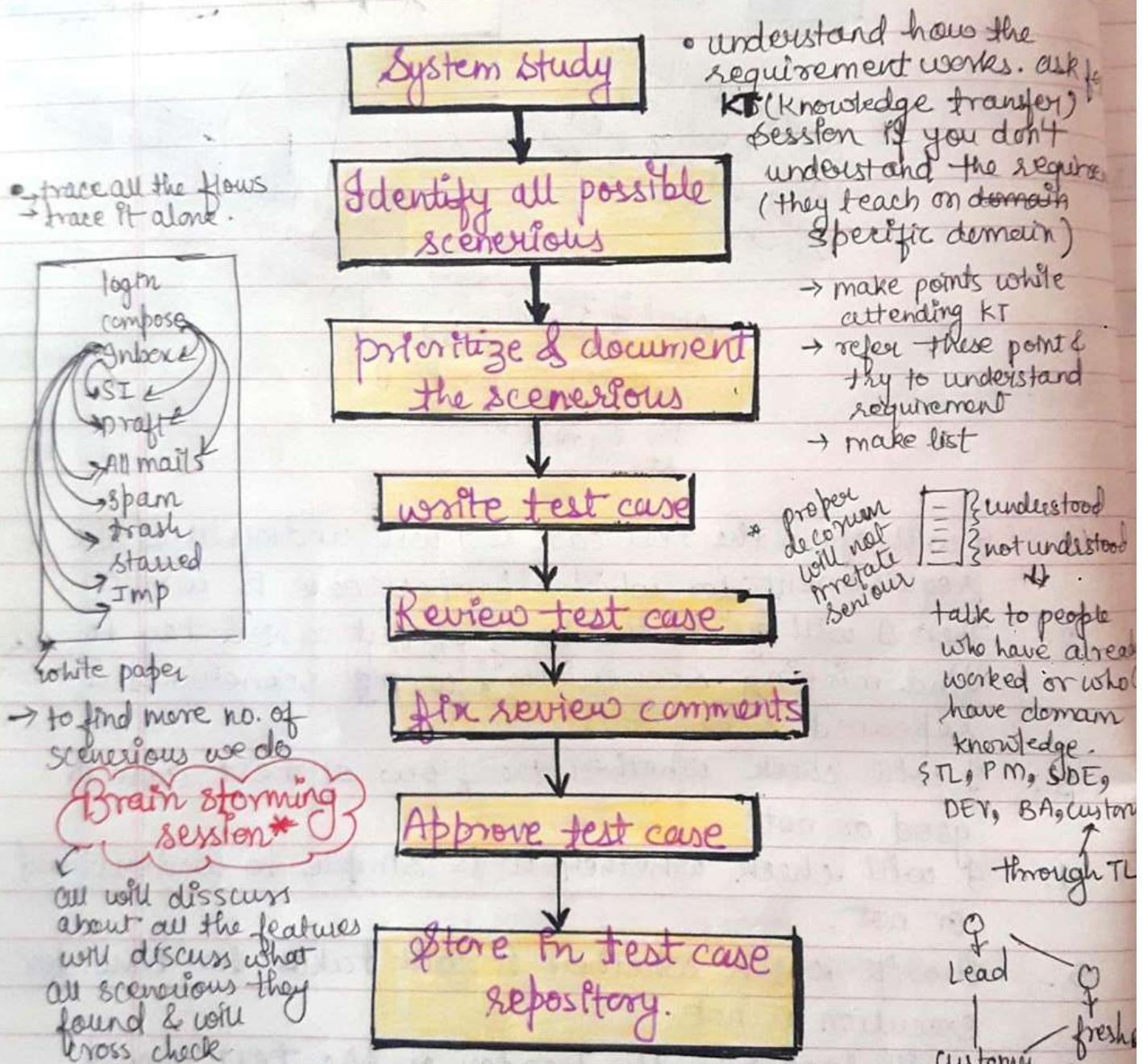
## Q. How do you review Testcase?



1. I will open the test case & I will understand the requirement for which the test case is written.
2. Then I will go to the body of test case & try to find missing scenarios, wrong scenarios & repeated scenarios.
3. I will check whether the flow of test case is good or not.
4. I will check whether it is simple to understand or not.
5. I will check whether it will take less time for execution or not.
6. I will look into the header of the test case & check whether all the attributes are covered or not.
7. I will check whether all the attributes in the header is having relevant data or not.

8. I will check whether template of the test case used is according to standard defined in the project or not.

## Procedure to write test case :

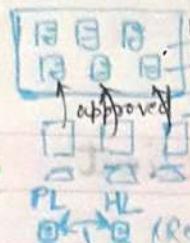


\* While identifying the scenarios we do brain storming meeting. It is a meeting conducted by testing team where they discuss all the features & find all the missing, wrong, repeated scenarios, & try to ↑ the no. of scenarios & also prioritise them.

repository → storage place / Database

here we store approved test cases

D:\ Test case repository



while testing the build TE take the test cases from test case repository

build

Test Server

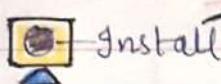
(Reviewed)

## Types of Application:

- you
- Work on →
- either either
  - 1). stand alone / Desktop based application / windows application
  - 2). client / server application.
  - 3). web based application.

1eg. Notepad++, calc, autocad, ms-office, paint

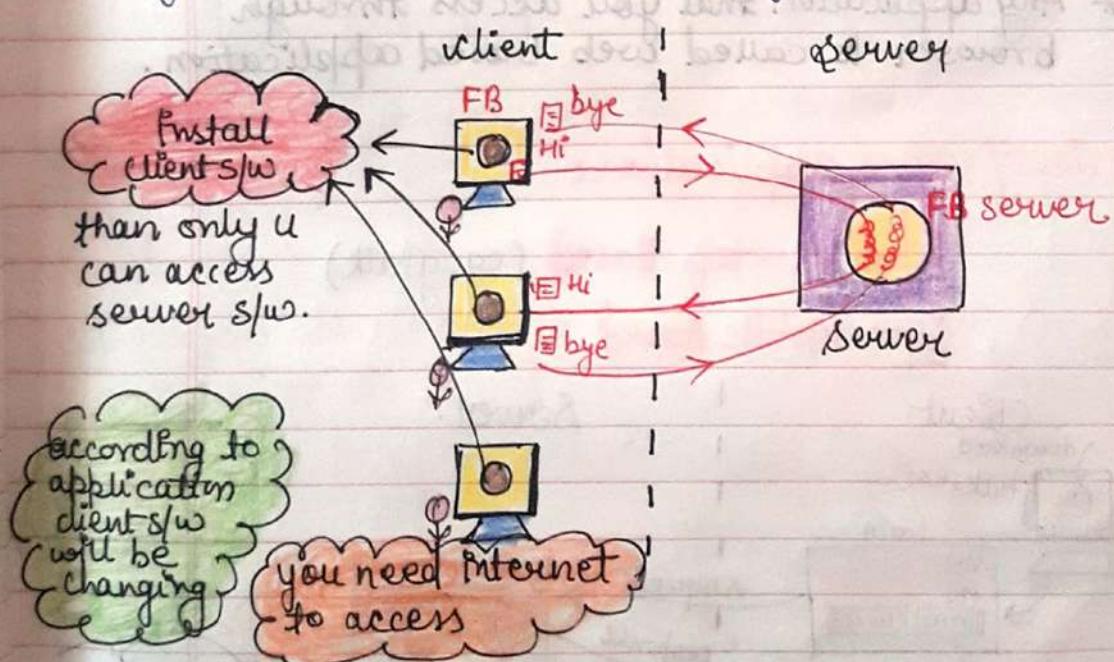
↓ Download



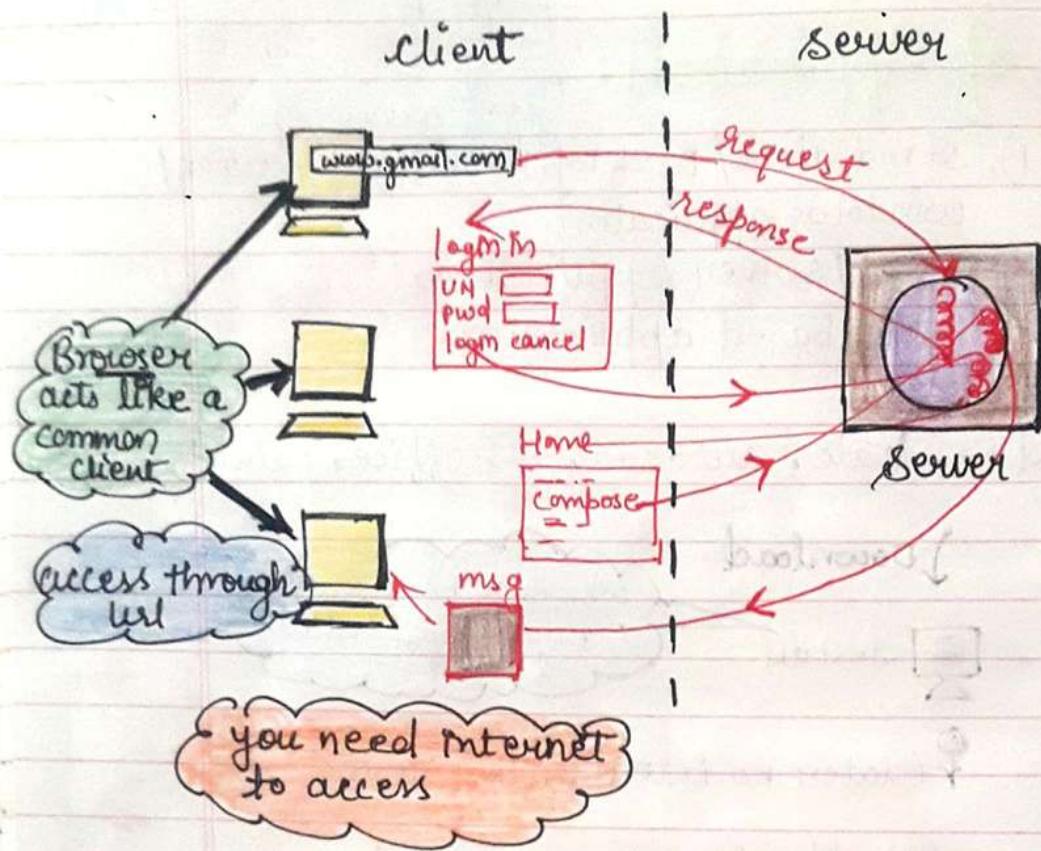
who ever need it, download install & use it

(later no Internet is needed)

2eg. ola cab, whatsapp, instagram, skype



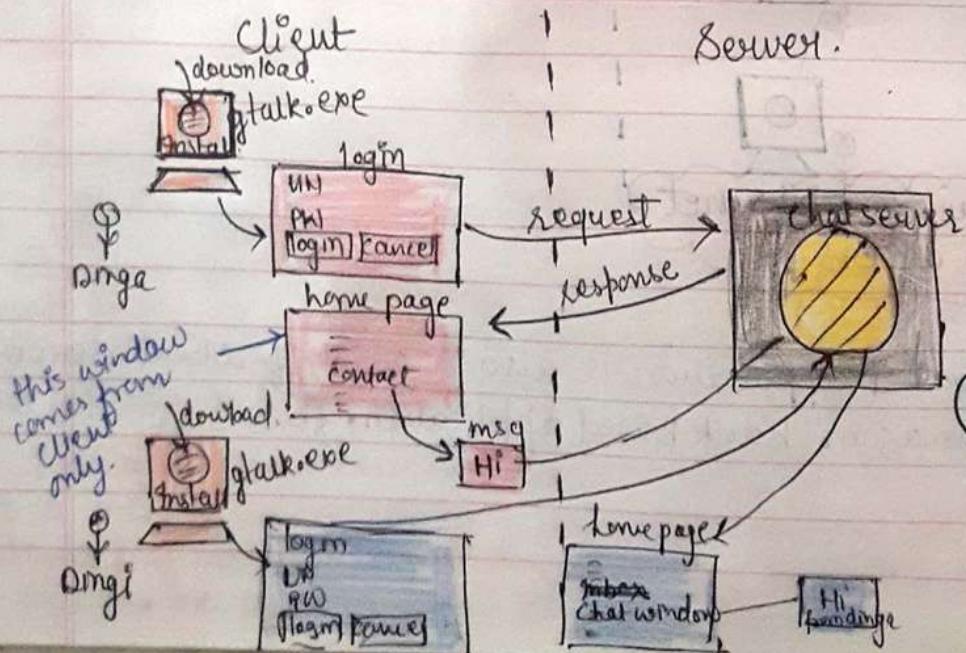
3eg. web based application is also a part of client-server app.  
- amazon, bank based application, fb.



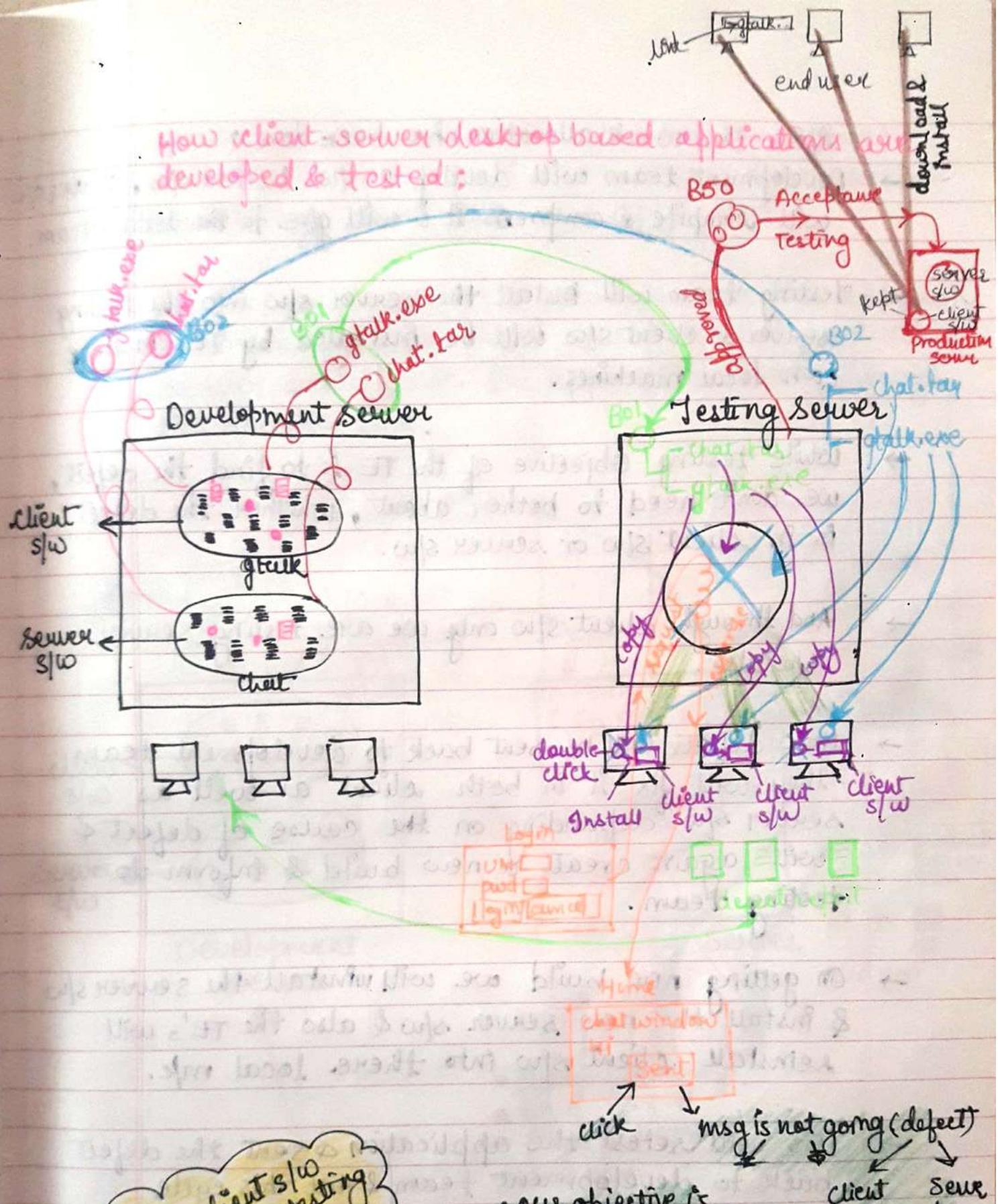
- \* Any application that you access through browser is called web based application.

### Client/Server application:

T → Desktop Based (eg. gtalk)  
 T → Mobile Based



How client-server desktop based applications are developed & tested : B50 Acceptance



through client s/w  
you are also testing  
server s/w also

- our objective is to find defect only.

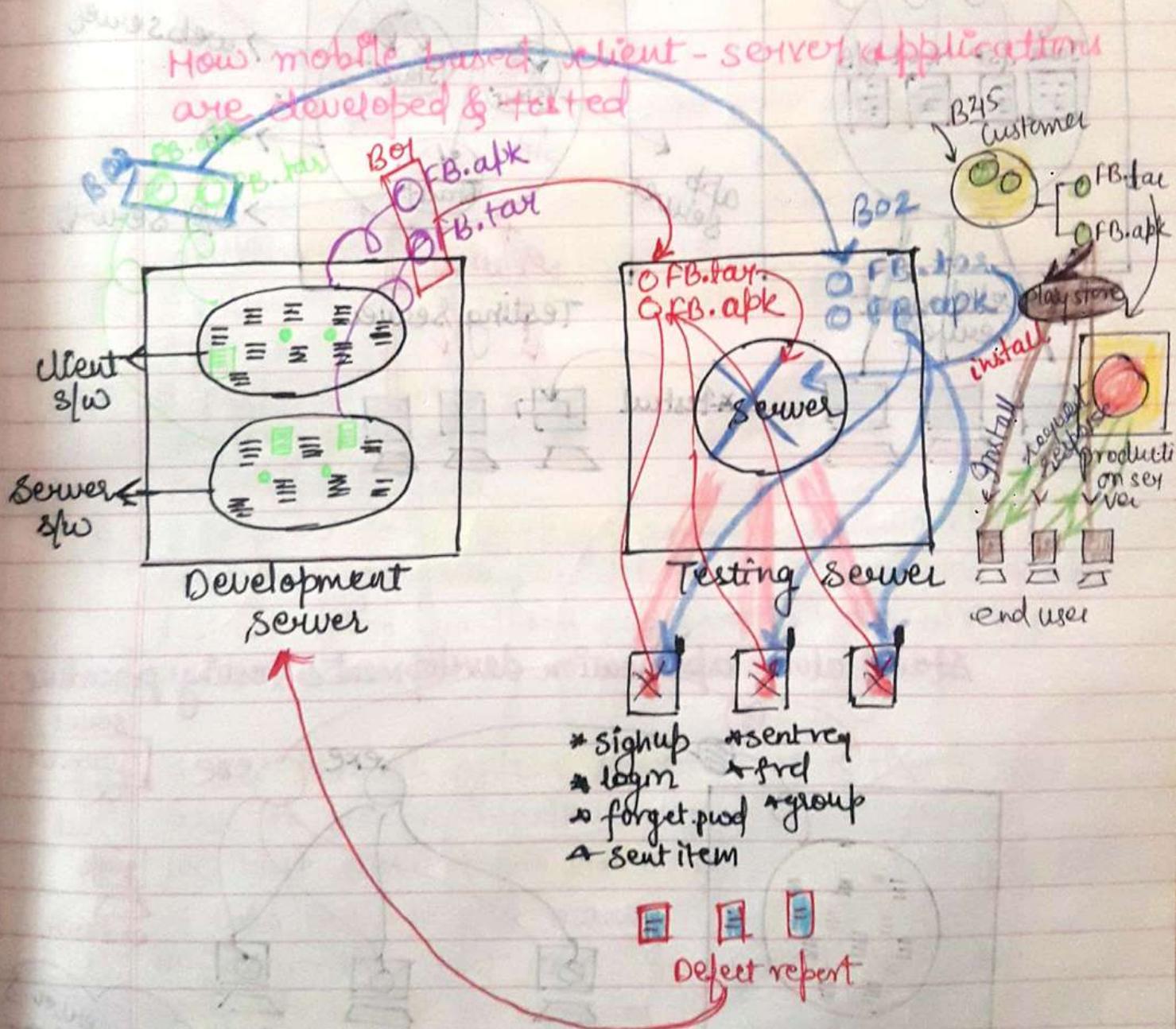
msg is not going (defect)

- u don't have to bother whether PIs client defect or server that devl - oper will understand & resolve .

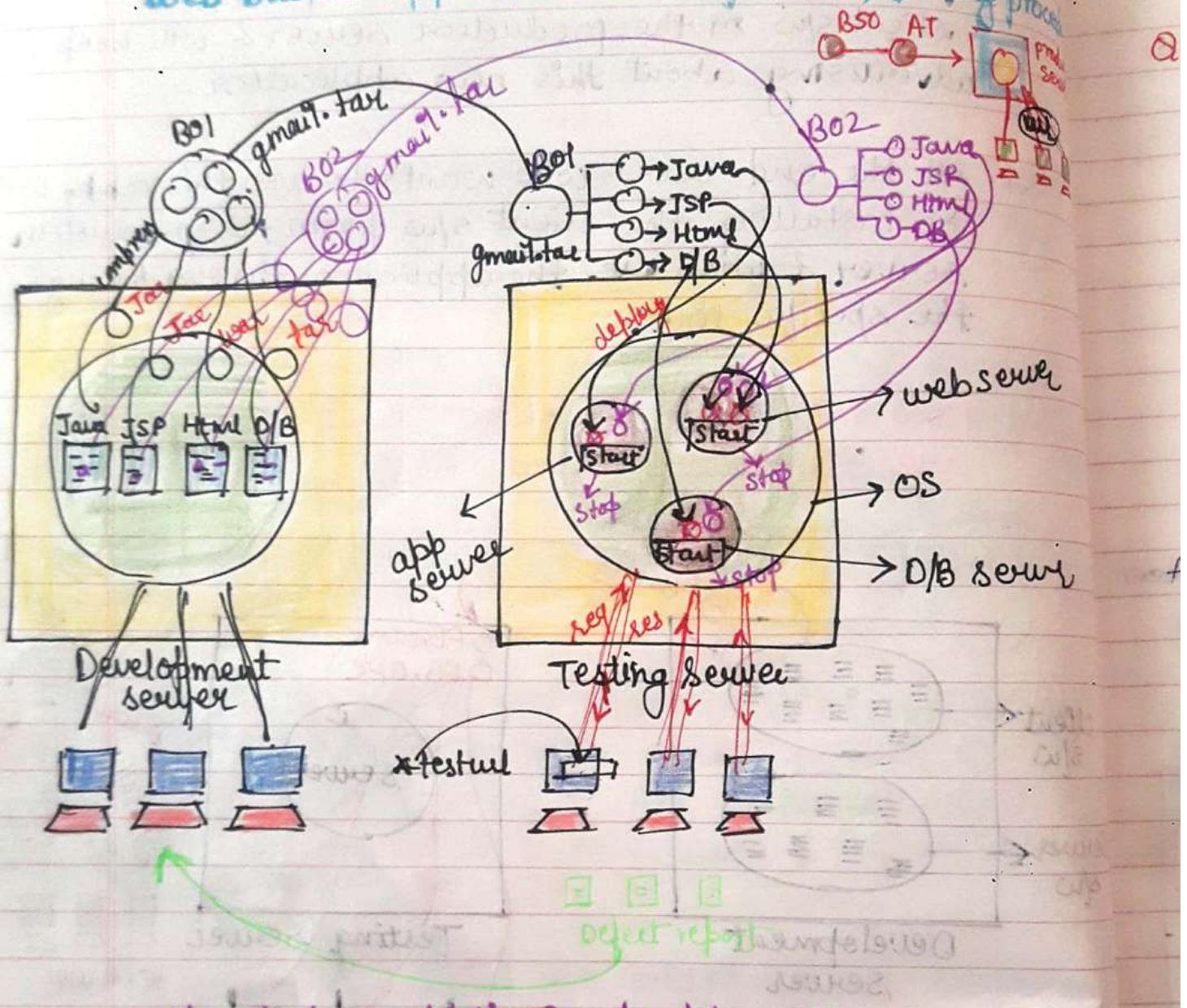
- After the work allocation has been done :
- Development team will develop 2 s/w's (client s/w, server) will compile & compress it & will give to the testing team.
  - Testing team will install the server s/w into the testing server & client s/w will be installed by TE's in their own local machines.
  - While testing objective of the TE is to find the defect, we don't need to bother about, whether the defect is in client s/w or server s/w.
  - And through client s/w only we are testing server s/w also.
  - Now defects will be sent back to development team they will fix it in both client as well as server s/w depending on the cause of defect & will again create a new build & inform to testing team.
  - On getting new build we will uninstall the server s/w & install the new server s/w & also the TE's will reinstall client s/w into there local m/c.
  - TE's will retest the application & send the defect back to development team & till this cycle continues until the s/w becomes stable.
  - For suppose B-50 is the stable build, then it will be given to customer & he will perform acceptance testing & will install client s/w & into

production server & will also keep (store) client sw in the production server & will keep on advertising about this new application.

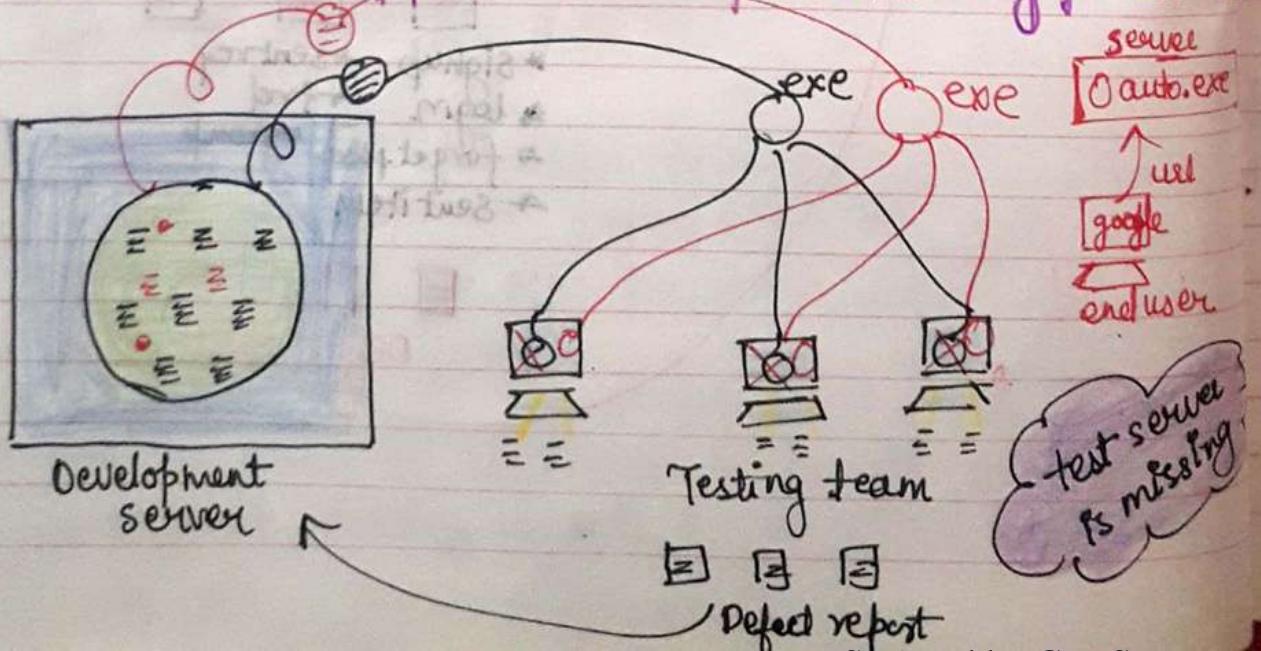
- All the end users who want's to use this app, will be installing the client s/w from this production server & will use the application through the specific link.



## Web Based application development & testing process



## Stand alone application development & testing procedure



Q. How do you convince ur manager that you have tested every thing?

(OR)

How do you ensure that ur test coverage is good?

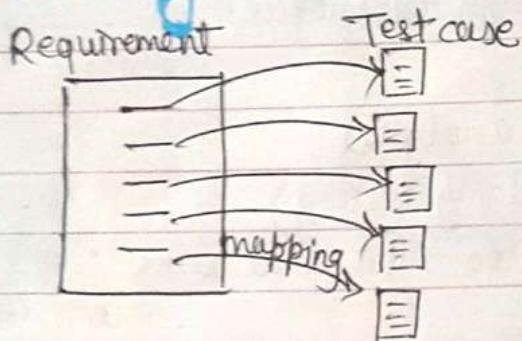
- followed strict procedure to write test case
- used test case design techniques.
- Traceability matrix
- CCA (code coverage analysis)
- adhoc testing (randomly testing)
- after getting build also I got some more scenarios.

Ans. My test coverage is good becoz I followed certain activities while writing test cases:

- ① I followed strict procedure to write test cases so my test coverage is good.
  - ② I prepared traceability matrix to ensure that every requirement has got atleast 1 test case so my coverage is good.
  - ③ I followed code coverage analysis technique & identified programs which are not covered while test execution for those program I have written the test cases.
- 
- a) I did thorough system study becoz of that I was able to derive maximum possible scenarios.
  - b) we have done brain storming session becoz of that I was able to find many missing scenarios.
  - c) while writing test cases I applied test case design technique because of that I was able to find accurate scenarios.
  - d) while doing test case review process, we were able to add missing scenarios.

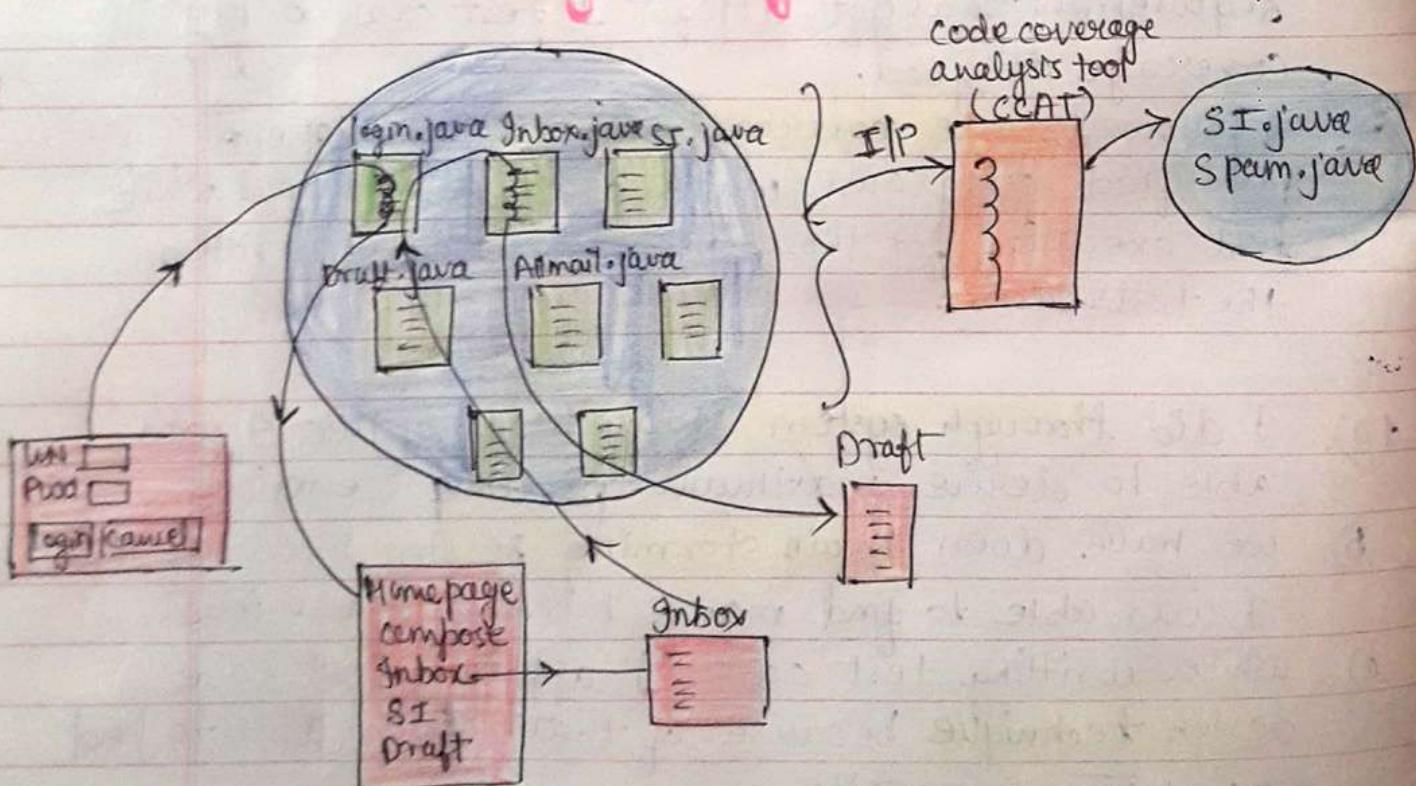
- e). while doing test execution I found new scenarios & added back into the test cases.
- f) I also performed adhoc testing because of that our test coverage is improved.

## Traceability matrix:



It is a document which ensure that each requirement has got atleast 1 test case & it is also k/s mapping the test case

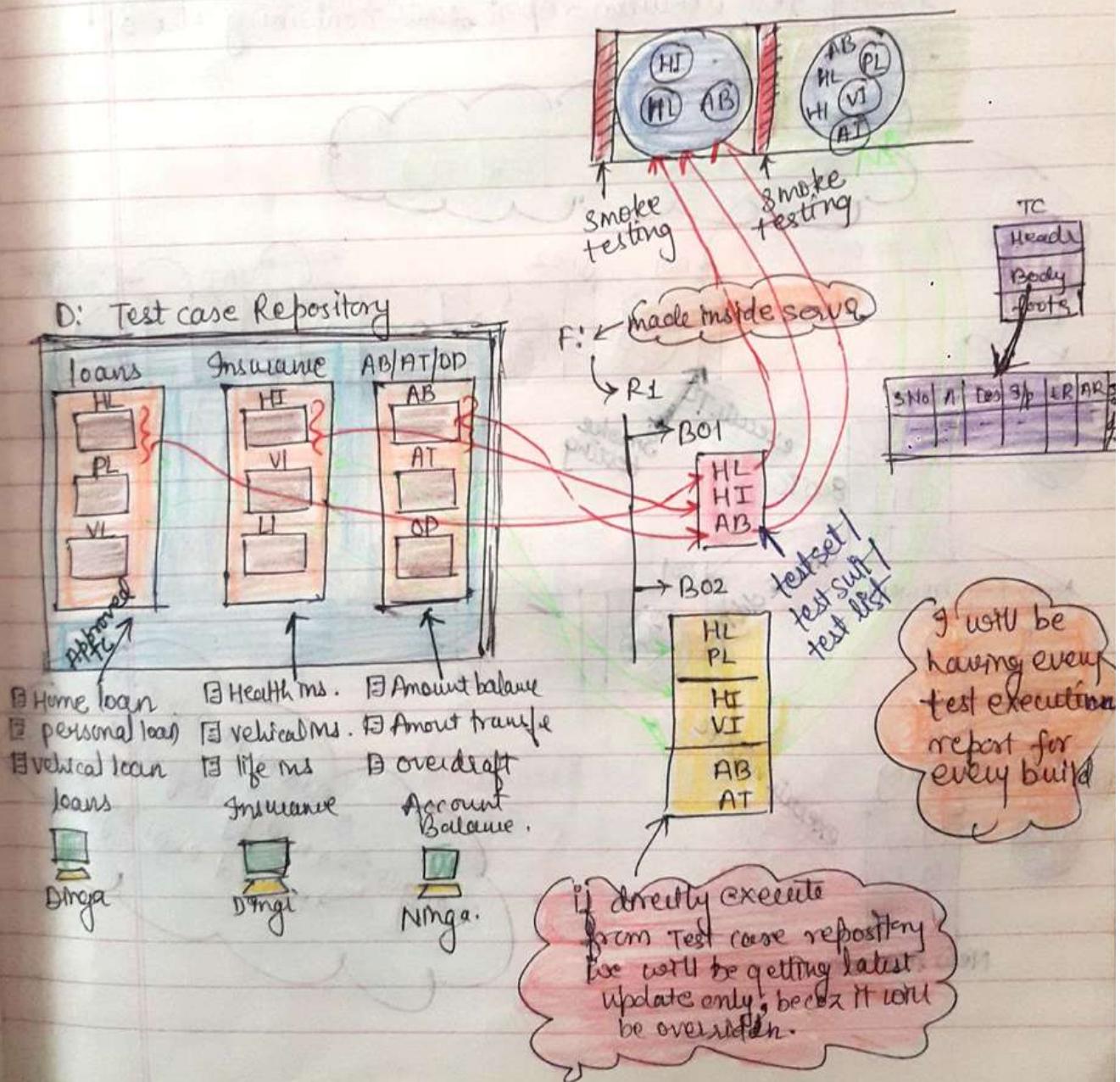
## Code coverage analysis:



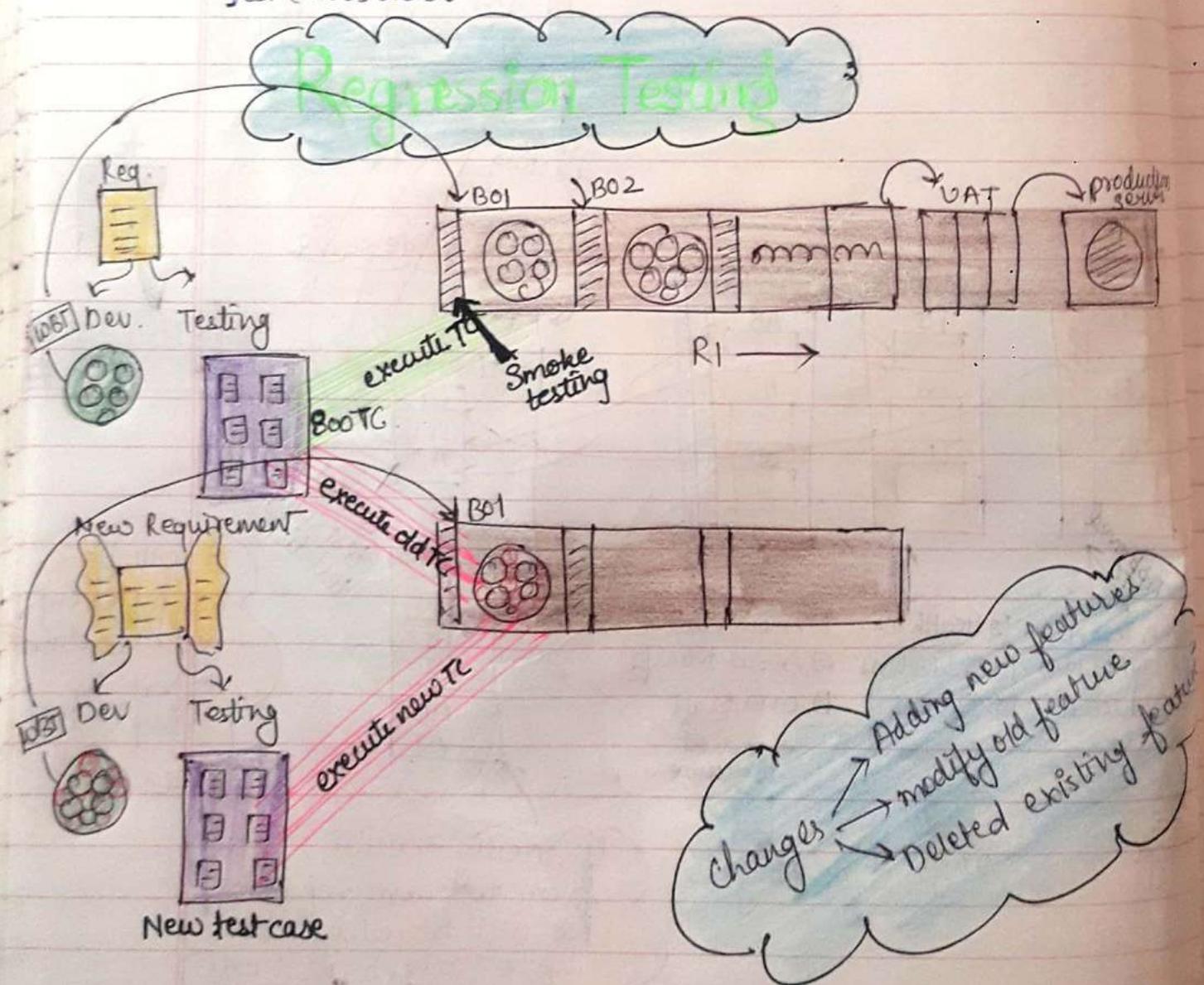
• testing old feature is called regression testing, by executing old test cases we do regression testing.

There might remain some program, which will not be executed & identifying it manually is difficult so we use coverage analysis tool, - this tool will list all the program while have which have not been executed. It may happen because we might have missed that test scenarios. So after doing code coverage analysis we may find some more missed scenarios.

## Test Execution Process:-



- All the approved test cases will be stored in the test case repository.
- Now when the developer give the build, tester will be copying the selected module test cases from the test repository and create test suit/test set/test list.
- Now tester will test the application by executing test cases & will fill the actual result & status field in that will send the report to developers regarding defect.
- This cycle will continue & for every build we will be having test execution report containing the old test cases also.



## **Definition:**

Testing the unchanged features to make sure that it is not broken because of changes is called regression testing.

The changes would be addition, modification, & removal of features or defect fixes.

(OR)

Re-execution of same test cases in different test cycles or releases to ensure that changes are not introducing the defects in unchanged features is called regression testing.

## **Types of regression testing:**

- Unit regression testing (Testing only the changes)
- Regional regression testing
- Full regression testing

### **I. UNIT REGRESSION TESTING:**

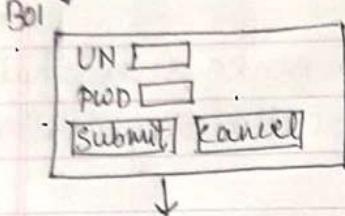
#### **Definition:**

Testing the changes or bug fixes is called unit regression testing.

#### **Note:**

When ever we are assured that changes will not affect any other features we go for unit regression testing.

eg1. ↓ create user



conf message  
(spelling mistake)

(spelling mistake)

} Development team

B02 (spelling mistake corrected)

It will not affect  
other features

egs

eg2.

↓ B01

user name

[ ]

8-10 chr.

↓ B02

user name

[ ]

4 to 16 chr.

egs

will test it again & it  
will not affect other feature

eg3.

↓ B01

[file view settings]

↓ B02

[file view settings]

new feature

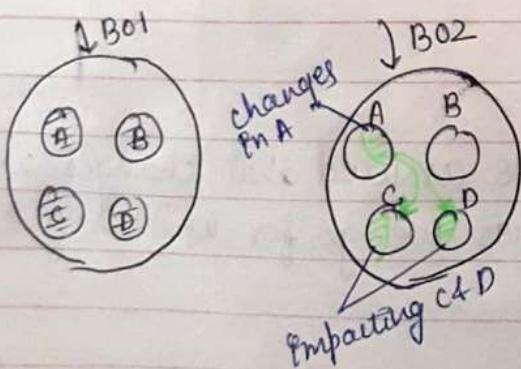
not regression  
testing.

## 2. REGIONAL REGRESSION TESTING:

Definition:

Testing the changes plus and Impacted areas is called as regional regression testing.

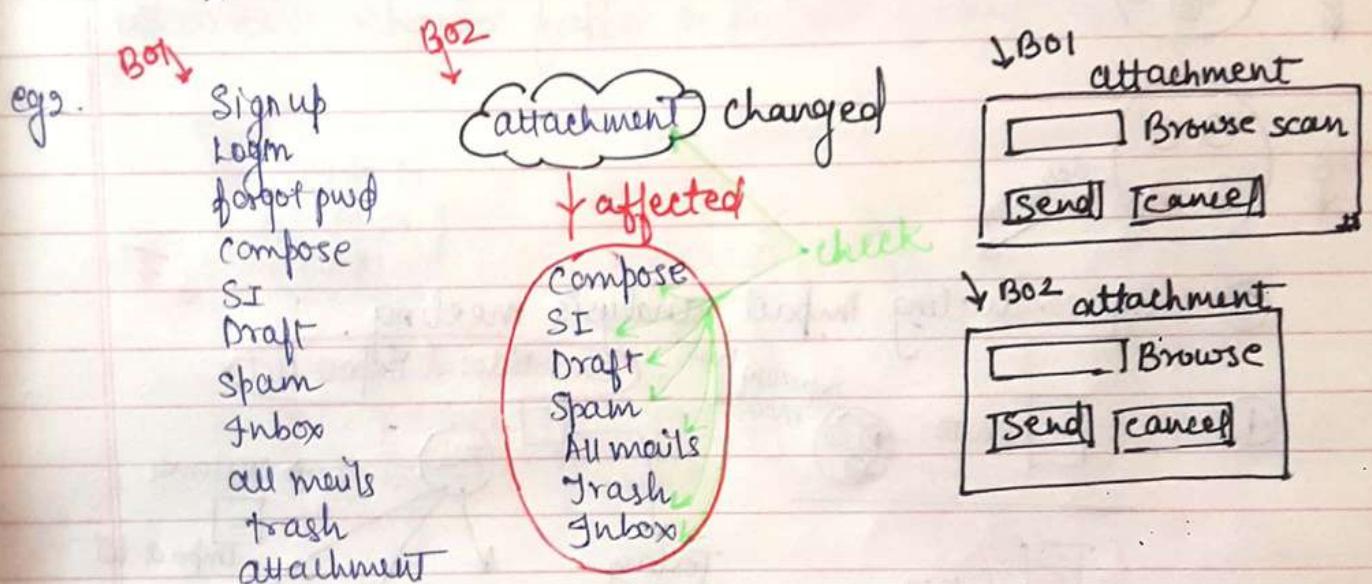
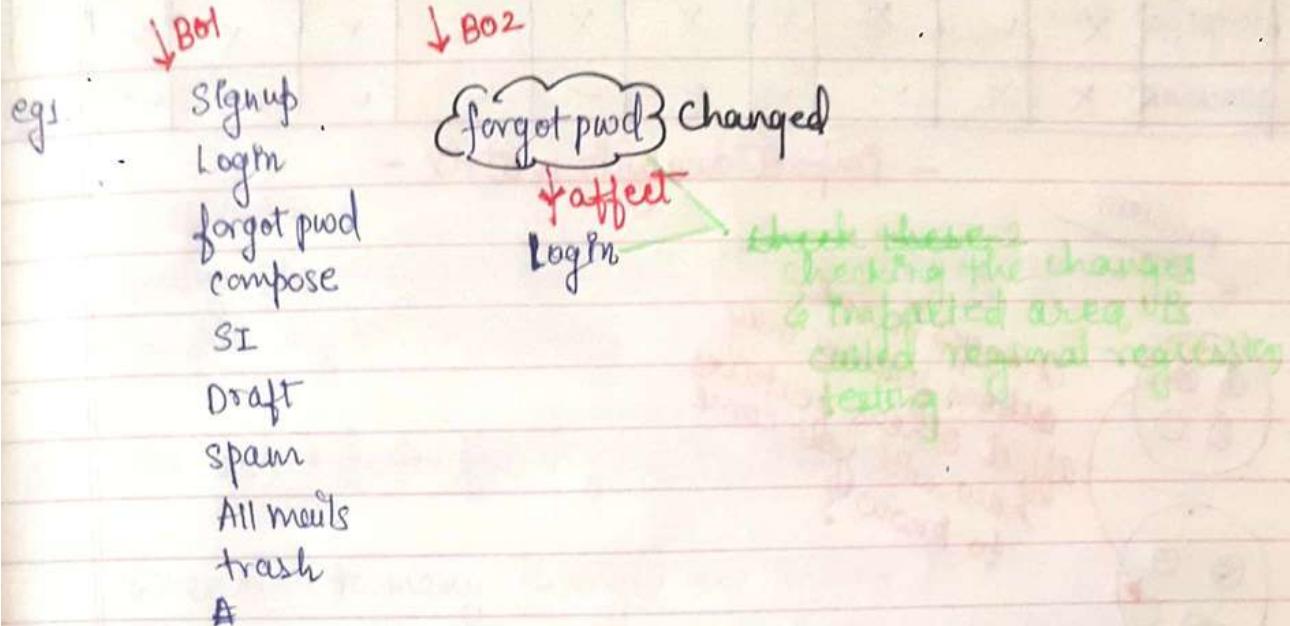
Q.



①

②

- based on domain knowledge we get to know impacted areas.



Q. How do you identify Impact area?  
(OR)

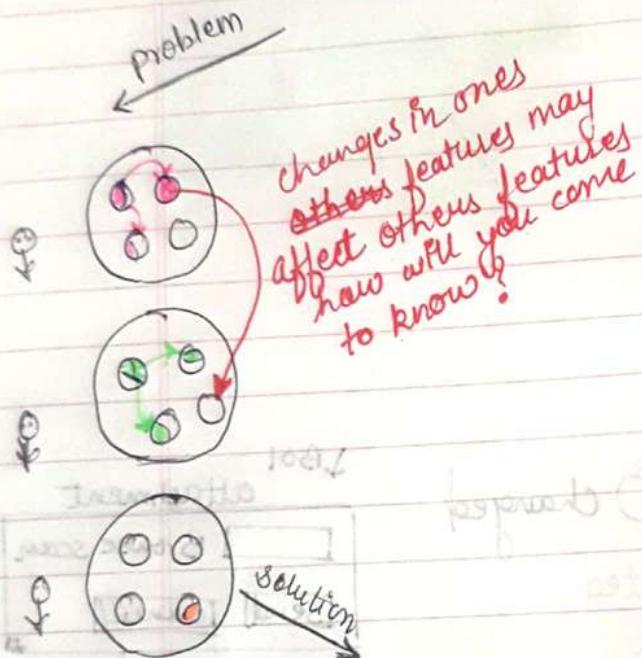
How do you do Impact analysis?  
(OR)

How do you pick regression test cases

- ① Based on domain knowledge: as a test er. I will be knowing how each & every module works & also how all the features are related, based on that knowledge I will be able to identify impacted areas
- ② By preparing Impact analysis matrix.

S.No.	Changes	Signup	Login	compose	SI	Draft	spam	All mails	trash	starred	group	cal
01	forgot p/w	x	v	x	x	x	x	x	v	x	x	x
02	attachment	x	x	v	v	v	v	v	v	v	v	x

### - Impact analysis matrix -



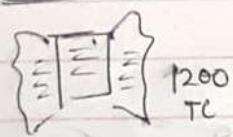
③ By conducting impact analysis meeting

consolidated impact list

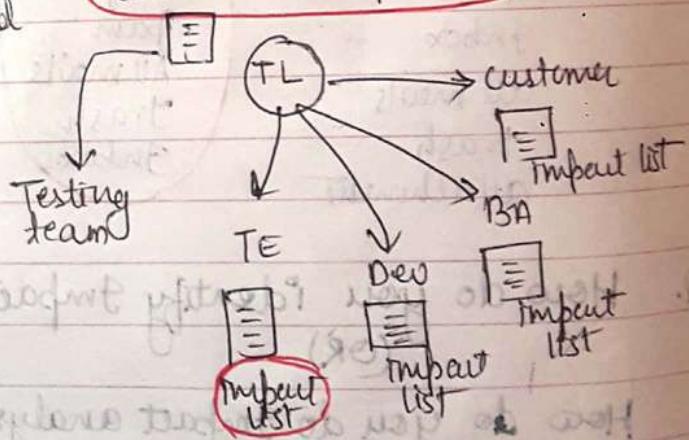
④ 800 TC



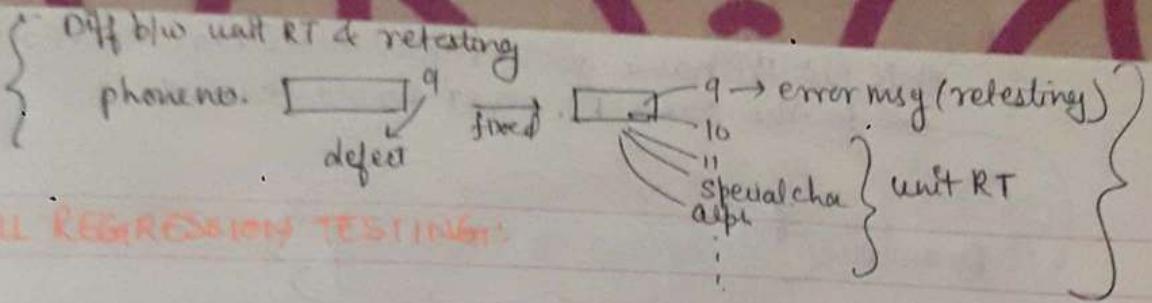
Impacting but missed



1200 TC



Communicating with Test lead, developer, BA & customer test lead will get the respective impact list & then he will make a consolidated impact list out of it & will give to testing team,



#### ④ FULL REGRESSION TESTING:

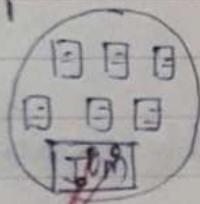
**Definition:**

Testing the changes and all the remaining areas is called as full regression testing.

#### Q. When do we go for full regression testing?

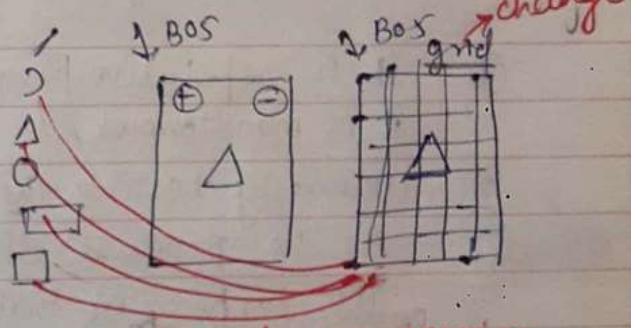
- ① Whenever too many changes are made in the project.
- ② Whenever changes happen in the root of the project.

e.g. 1



Changes in JVM,  
thus we need to  
do full regression  
testing.

e.g. 2. Autocad



Do full regression  
testing for all the  
features

- ③ Before giving the s/w to the customer, for last few cycles we do <sup>full</sup> regression testing.
- ④ After every few cycles we do regression testing in order to ↑ the quality of the product.

#### Q. What is the difference between regression testing & retesting?

Retesting: Check whether the bug is really fixed or not.

Regression testing: <sup>Testing</sup> whether the unchanged feature to make sure that it is not broken because of changes.

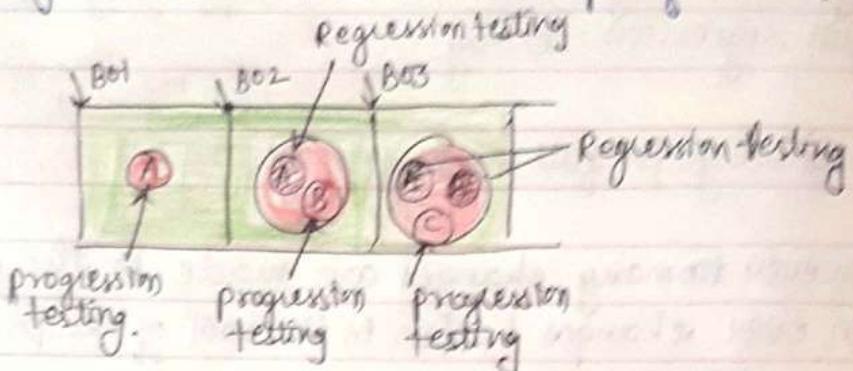
It is just the name given regression testing, In regression testing we do the same FT, IT, ST for old features.

Drawback

## Progression testing

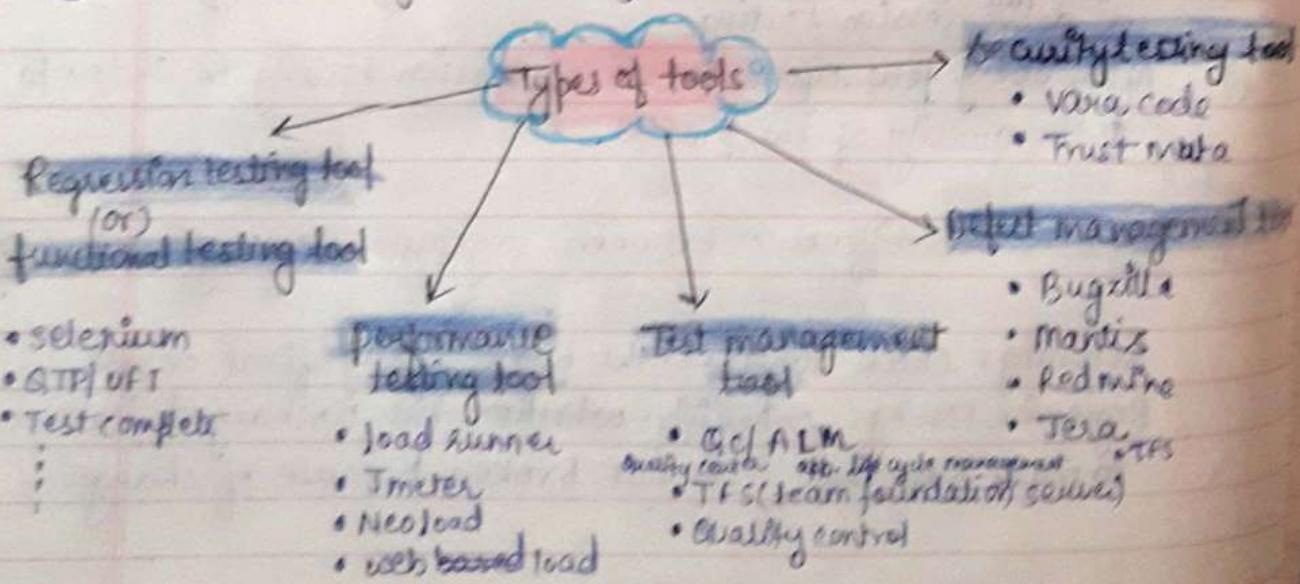
Definition:

Testing the new module is called progression testing.

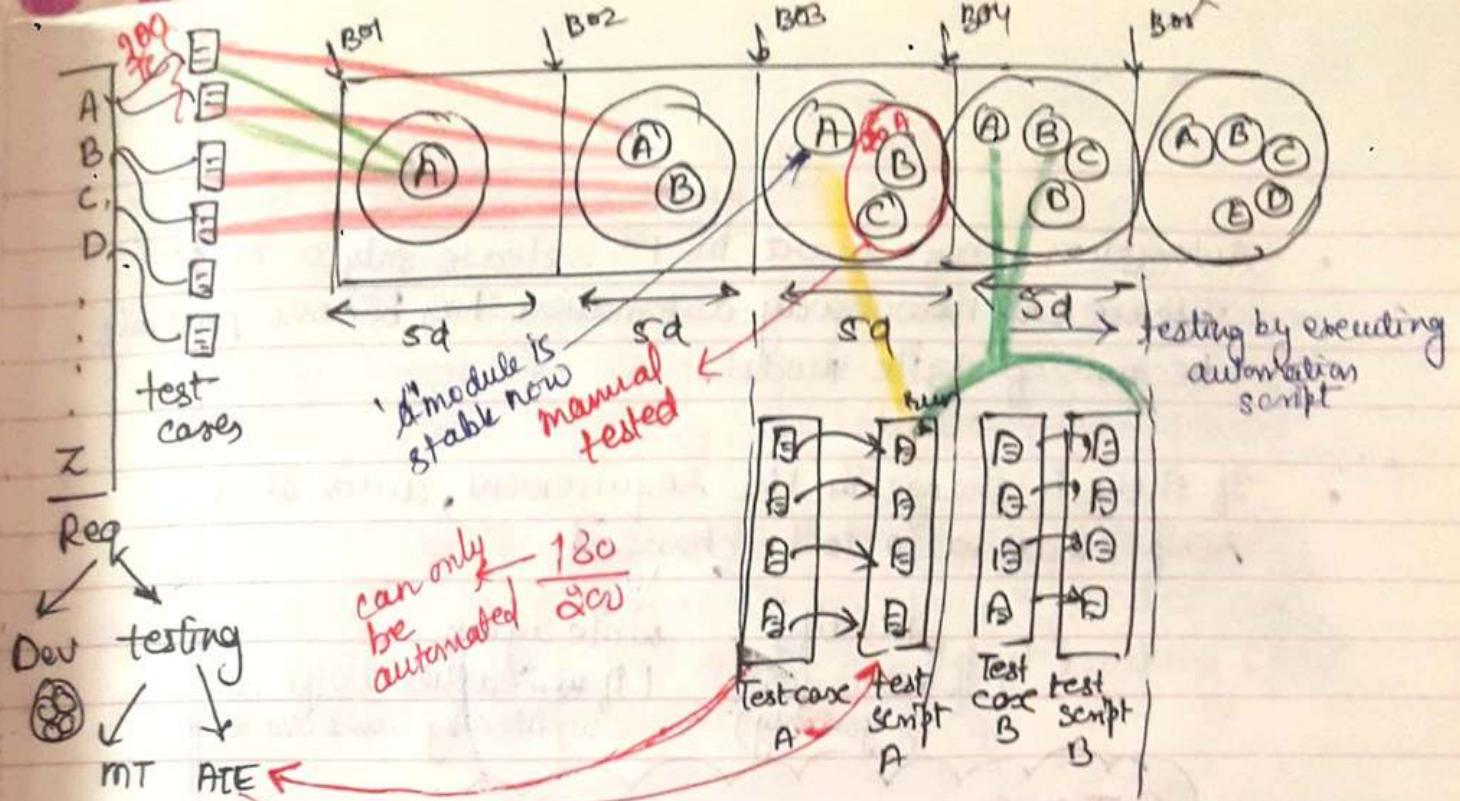


Drawback of manual testing:

- ① It is repetitive in nature over the period of time, it is monotonous.
- ② Manually testing the SW always consumes more time.
- ③ As the product size & the test cycle duration also because of that turnaround time will be more
- ④ It require more resources.
- ⑤ More human error.
- ⑥ No accuracy in testing



Time taken to deliver the proj to customer is called turn around time



- even automation script are ready we do 2-3 rounds of regression testing manually before delivering the S/w to customer
- Baseline:** comparing the result of automation testing with result of manual testing in order to test-test script
- Defect will be raised by manual team even when it is found by automation team otherwise these automation er. will get busy in testing (manually) on getting the fixed build rather than doing their work (i.e. writing test script)
- Defects found by executing the automation test scripts will always be less as it is ~~more~~ written after the s/w got stable.
- Some set of test cases can't be automated always like captcha, fingerprint, barcode reader (where manual intervention is required)

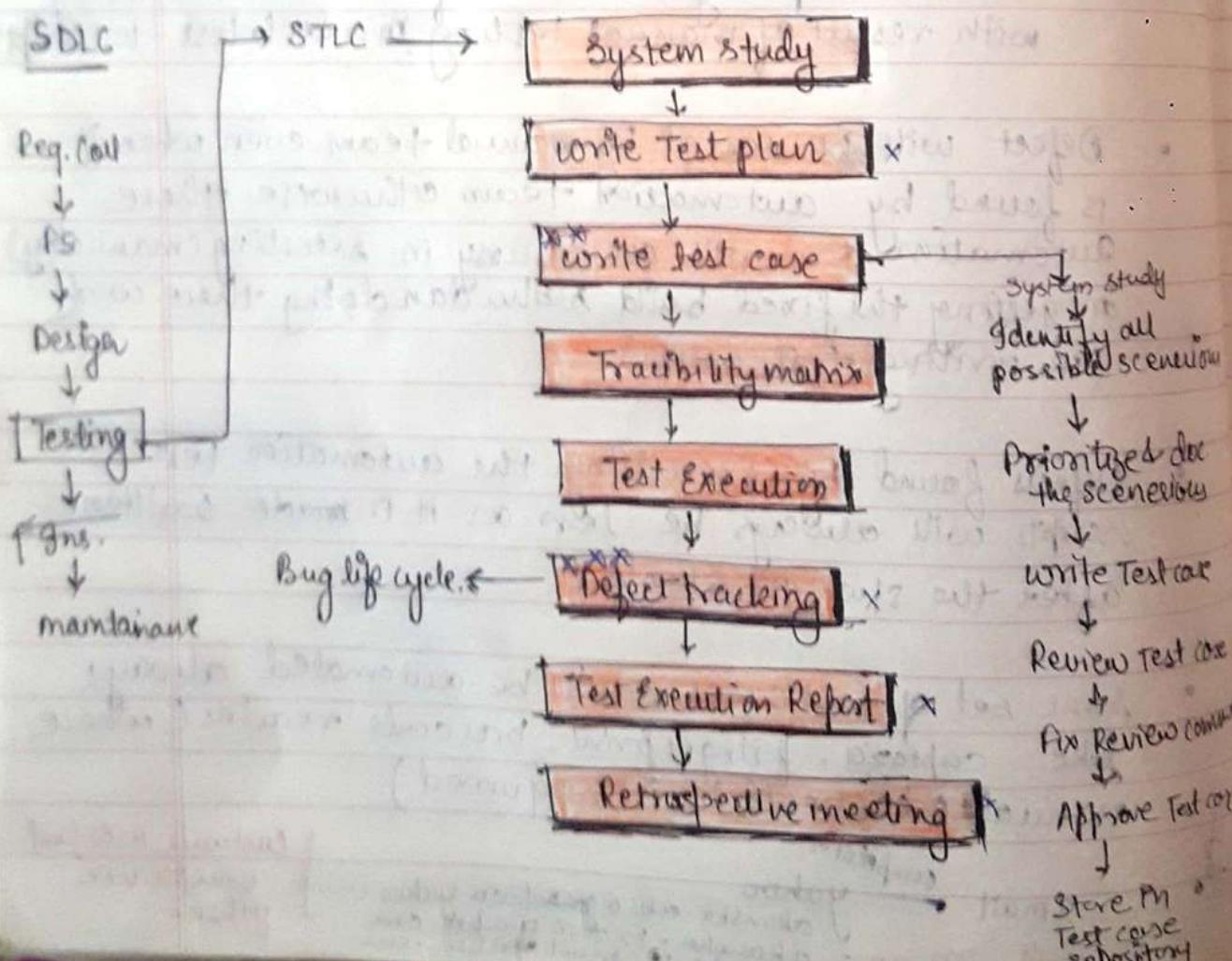
time. use of captch  
will create fake  
gmeet competitor yahoo  
akansha.ebc@gmail.com  
akansha.123@yahoo.com  
akansha.1@gmeet.yahoo.com

| customer will feel  
bored to use yahoo - !

- Automation can start in 1<sup>st</sup> release only or in 2<sup>nd</sup> release but now early automation has become possible because of agile model.
- If there is change in the requirement, automation script have also to be changed.
  - modify  
(if re-written by you only)
  - write again  
(if earlier script is written by someone else)

## STLC (sw test life cycle)

- It is the procedure to test the sw. It has got different stages.



Q. what is the diff b/w SDLC & STLC?

SDLC - It is the procedure to develop the SW & it has diff stages like req. collection, feasibility study, Design, coding, testing, installation & management.

STLC - It is the procedure to test the SW & it has diff stages like sys study, write test plan, write test case, traceability matrix, test execution, defect tracking, test execution report & Retrospective meeting.

### System study:

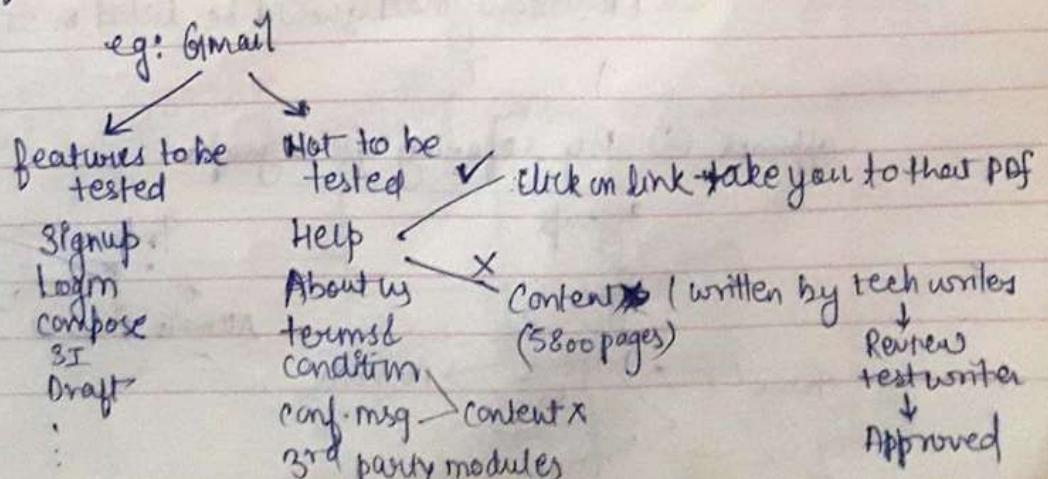
Here we go through the req. & understand the req. while understanding if we get any queries communicate with development team, business analyst & customer and get it clarify clearly.

### Test Plan:

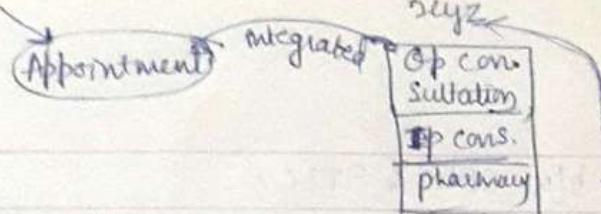
It is a document which derives all the future testing activities. It has got diff sections like

1. objective : This section covers aim of preparing test plan.

2. scope : This section covers list of features to be tested (in scope). This section covers list of features not to be tested (out of scope)

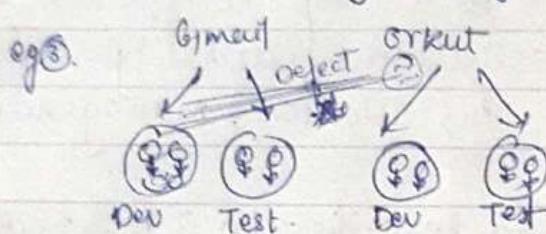


3<sup>rd</sup> party modules eg① ABC

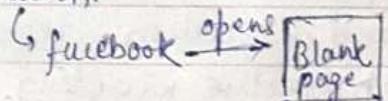


- will not check appointment module
- check only whether it is integrated properly or not

eg② online shopping → payment



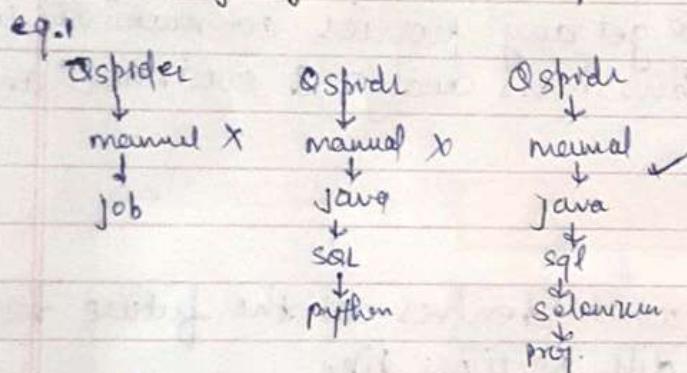
eg③ Application.



You will raise this defect (because link is not given properly)

3. Approach: This section covers, how we are going to test the product in future.

to get job



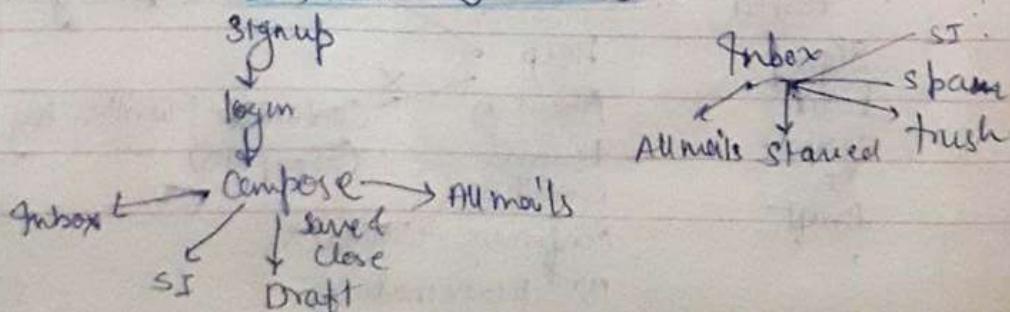
eg2

Approach By referring high level scenarios.

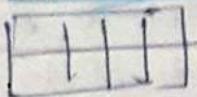
- ① login to gmail
- ② click on Compose link
- ③ enter all the data & click on Sent button
- ④ click on inbox link.

ER: Sent mail should be listed in inbox page.

approach ② By referring flow graph



approach③. By referring detailed test case.



approach 3 is good, if long term project

approach 2 is good, If short -terms project

if it will be sent to customer & he will approve  
which approach to follow.

#### 4. Test methodology :

This section covers what are the types of testing we should conduct in future.

Web based application:

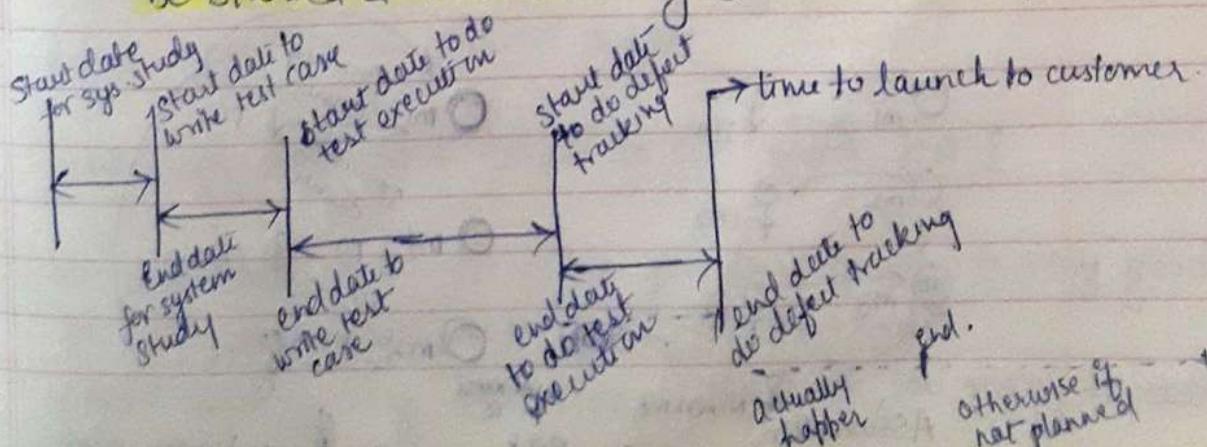
Smoke testing, FT, IT, ST, Regression testing, UAT, performance, usability, compatibility, acceptance, Adhoc, exploratory, globalization, security, recovery.

Stand alone application:

Smoke testing, FT, IT, ST, Regression testing, UAT, usability, compatibility, acceptance, Adhoc, exploratory, globalization, recovery, installation & uninstallation.

#### 5. Schedule :

This section covers when exactly which activity should be started & which activity should be completed.

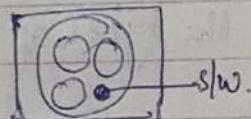


⑥ Test Environment: This section covers what are the H/W & S/W to be used in order to set up test environment. This covers procedure to install the S/W.

① procedure to install the S/W

= }  
yet not build.

② @ Server side.



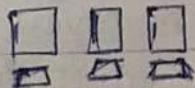
software:

OS?  
webdriver?  
app server?  
DB server?

H/W:

=

③ Client side



Software:

OS - W10

Browser - Mozilla, chrome, opera

MS-Office

=

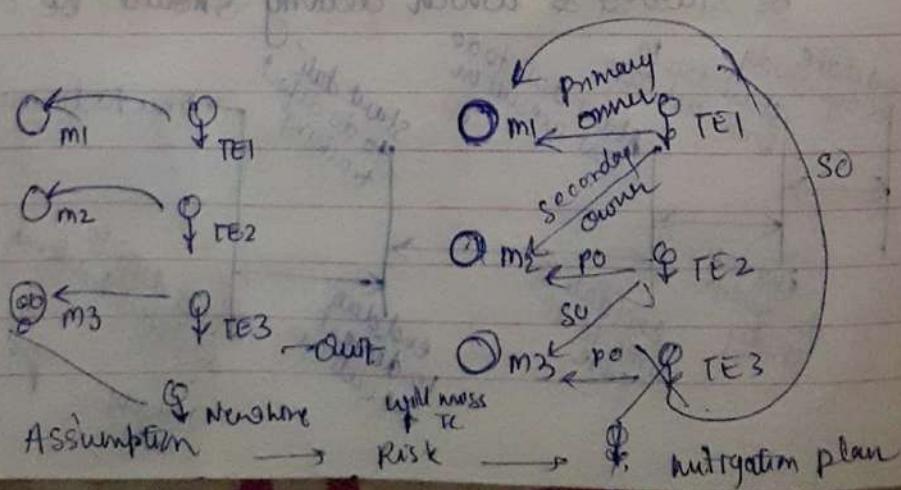
Hardware:

= } 2 GB Speed  
16 GB RAM.  
10 TB.

⑦ Assumption

⑧ Risk

⑨ Mitigation plans / backup plan



**assumption**: all the TC's will be there till the end of the proj.

**Risk**: If some one quit chances are there new Er. might miss lot of defects

**Mitigation plan**: New Er. should take help from secondary owner & test the application.

### ⑩ Deliberables:

This section covers the o/p of testing team to the customer by the end of the project.

(test case document, test plan, traceability matrix, test execution report, graphs & metrics & release note).

Release note - new features add, modified features, defect fixes, pending defects, demo of the current release, removed features.

of previous release

Q. Why we get critical defects in the <sup>n<sup>th</sup></sup> moment of the SW?

- ① Because of adhoc testing, test Er. would have got creative scenarios at the end of the project.
- ② Test Er. would have neglected in the beginning & finding at the end of the project.
- ③ Because of <sup>n<sup>th</sup></sup> moment changes, done by the developer that might introduce few critical defects.
- ④ Test Er. There are the chances fixing the major or minor defect might create critical defect.

Q. Test stop criteria or when do we ~~stop~~ <sup>stop</sup> test?

- ① We stop testing when ever the product quality is very good or when ever the product quality is very bad.  
product quality very bad means there are too many blocker

& critical bugs, when end-to-end critical business scenarios are not working.

Product quality is very good means all the end-to-end scenarios are working fine, where there are no blocker & critical defects, there might be few bugs left but they are major or minor & it is less than the acceptable limit set by the customer.

- ② When it process the budget & if it is crossing the deadline of the schedule.

## ⑪ Defect tracking :

This section covers how we should track the defects in future. It involves procedure to track the defect.

### 1) Procedure to track the defect

- ≡ \*
- severity {
- \* priority }

### 2) Severity levels.

- \* Blocker (or show stopper)
- \* critical
- \* major
- \* minor

### 3) Priority level

- \* urgent
- \* high
- \* medium
- \* low

### 4) Defect tracking tool to be used

Bugzilla / mantis / selenium / zentao / TFS

## ① Effort estimation:

This section covers total estimation of the project.

$$\text{Total man hours.}$$

$$\rightarrow 5000 \text{ hr}$$

$$5000 \times x \$$$

$$= z \$ ?$$

$$1 \text{ hr} = x \$$$

Done by PM

$$\cdot \frac{5000}{8 \text{ hr}} = 625 \text{ days}$$

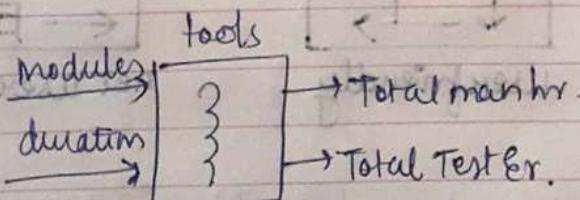
Customer wants in 3 months ie  $22 \text{ days} \times 3 = 66 \text{ days}$

$$\frac{625 \text{ (total days)}}{66 \text{ (estimated days)}} = 10 \text{ TE}$$

How to find total man hrs?

SLNo.	Module Name	Activity	System Study	Identify Scenarios	Prioritised document Scenarios	Writing Testcase	Review Test case	Testability review	Total Est. time	Total hrs
01	Compose	2 hrs	-	-	-	-	-	-	-	-
02	Index	-	-	-	-	-	-	-	-	-

total: 3000  
estimated.



Knowledge  
previous

3000 ← 5000

+ 30%

6500

← 9000

Learning, ← 3000 ← 3000 ←

# Usability testing

- non-functional testing.

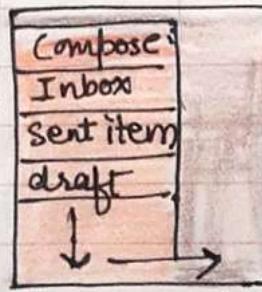
## Definitions

Testing the userfriendliness of an application is called as usability testing  
(OR)

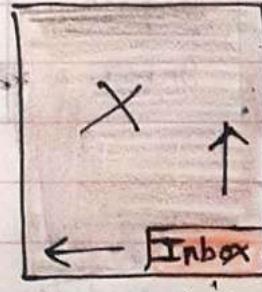
The effort spend less on an application to use it is called as userfriendly application. Testing this is called as usability testing.

On what bases usability testing is done?

- ① All the frequently used features has to be from top to bottom & from left to right. being a test er. we list out all the frequently used features of an application & check if it is from Top to bottom & left to right.

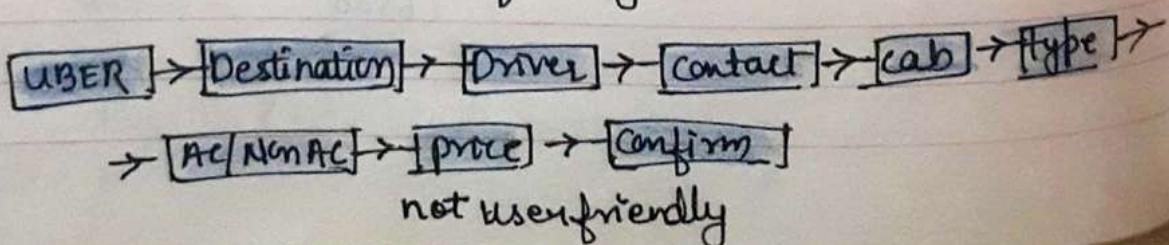
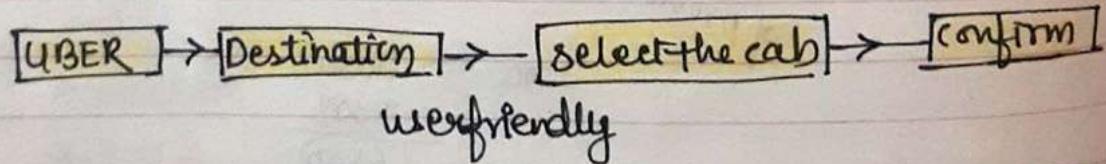


userfriendly



not userfriendly

- ② No. of actions performed on the application has to lesser than 5 clicks.



③ Easy to understand;  
The app has to be simple & it can be used by any end users (frequent users & new users)

④ Look & feel of the application:

The application used by the end user is always judged based on the look & feel of the application (pleasant to your eye).

For what applications we do usability testing?

- ① If the application is expected to be very simple & can be used by any end users, then on such applications we do usability testing.
- ② If the application does not require any training than we do usability testing on such application.
- ③ If the application generates a lot of revenue than we make sure we do usability testing (e-commerce application, banking app.)
- ④ If the customer is keen on userfriendly of the app. than we do usability testing of the application

Who does usability testing?

It is a good approach if the end-users does usability testing on the app. rather than TE's doing it, because by the time TE's performs various tests on the application the app. becomes userfriendly to them.

## How to do usability testing?

- ① Identify end users who frequently use the application & who is the new user.
- ② Provide the check list to both the end users  
eg. Flight booking application

Activities	simple to understand	look & feel	No. of actions
Search			
Book			
Status			
Cancel			

checklist

5 - Very good  
4 - good  
3 - Average  
2 - Bad  
1 - poor

- ③ Ask the end users to rate the application/modules based on simple to understand, look & fell & no. of actions.
- ④ collect the checklist from the end users & finally consolidate the features which has got the rating 3 & lesser than 3.
- ⑤ sent it to the development team raising it as a bug & asking them to make it more userfriendly.

when to do usability testing:

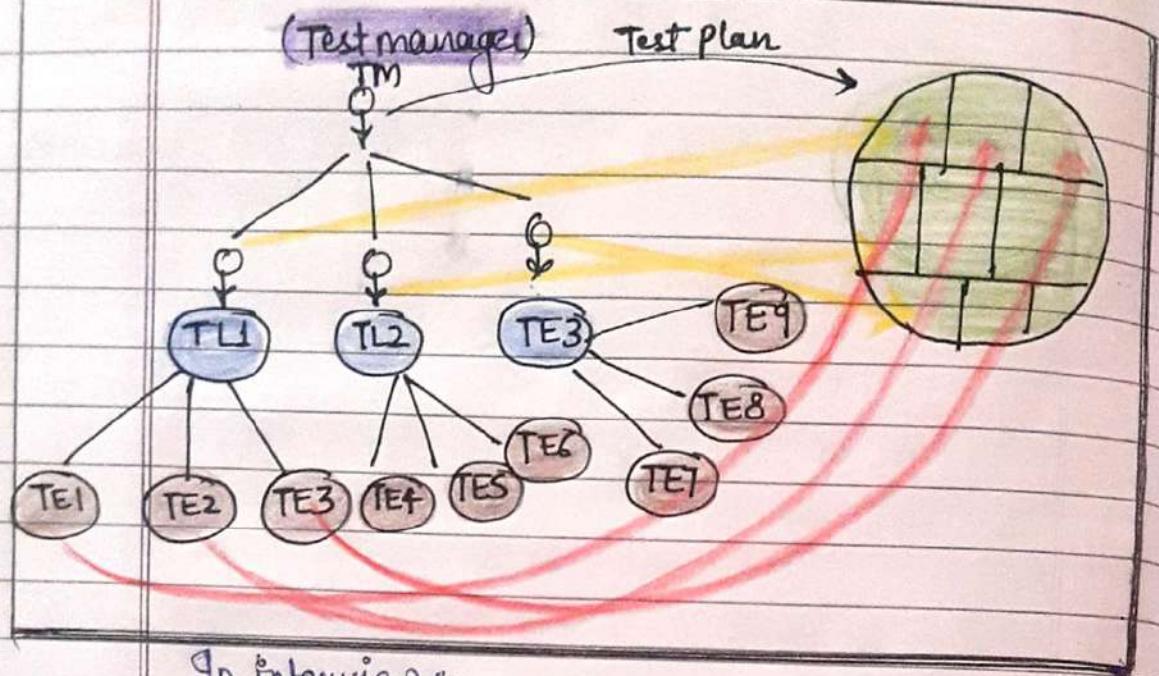
- ① When the customer is very keen on the userfriendliness of the app. we will do it from the initial build onwards.

⑨ If the customer is not very keen on the userfriendliness of the app we will do at the end when all the possible testing has been done.

### Assignment

Prepare a checklist for makemytrip.com, Yatra.com & cleartrip.com on activities search, book, status & cancel.

### 13. Roles & Responsibility



In Interview :-

Test plane ~~was~~ is written by  
60% → TM  
40% → TL

But actually :-

60% → TL  
20% → Senior TE  
20% → TM

Test manager define test life cycle & ensure that people are following it

- ② He prepare test plan.
- ③ Interact with developer, customer & management
- ④ Handle issues & act as mediator.
- ⑤ Effort estimation.

### Test Lead:

- ① Assign the work to test Er & ensure that they complete the task within the schedule.
- ② Consolidate the report sent by test Er & send it to customer, developers & management.
- ③ Write / review test plan.

### Test Er:

- ① Review test plan.
- ② Write test case
- ③ Review test case
- ④ Execute test case
- ⑤ Perform diff types of testing on the product
- ⑥ Involved in the test execution, diff tracking
- ⑦ Involved in preparing test execution report.

### Automation test Er:

- ① Setup the test Env. & Install the product
- ② Identify the test cases to be automated.
- ③ Automate all the regression test cases

#### ④ Execute & maintain automation test script.

#### 14. Templates:

This section covers templates for all the documents which we are planning to prepare in future

- ① Test case format template
- ② feasibility matrix
- ③ Test execution report
- ④ Defect report
- ⑤ Graphs & matrices.

#### Q. Who write test plan?

Test lead or test manager

#### Q. Who reviews test plan?

Test lead or test manager, development team even customer can also review test plan

#### Q. Who approve the test plan?

Test manager or customer approve the test plan.

#### 15. Test automation:

- ① This section covers which are all the test cases can be automated.

- ④ which are all the test cases can't be automated
- ⑤ which automation tool to be used.

### Traceability Matrix:

It is a document which ensure that every requirement has got atleast 1 test case.

SL No.	Module Name	High level req.	detailed req.	Test case Name	Automation Script Name
01	Locans	HL	R1.1	TC1	TS1
			R1.2	TC2	TS2
		PL	R2.1	TC3	TS3
			R2.2	TC4	TS4
		ML			

Some items Traceability Matrix

### Advantages of traceability matrix:

- ① It gives traceability from high level requirement till automation test script
- ② If suddenly requirement change, we will be knowing which exact test cases & automation test script to be modified
- ③ We can ensure that every req. has got atleast 1 test case.

## Drawbacks of traceability:

① It will not ensure we got 100% coverage.

e.g. 1 Req → Should have 15 scenarios → but u have written 10

Q. What is the diff. b/w traceability matrix & test case review?

In traceability matrix we check whether every req. has got at least 1 test case

In test case review we check whether all possible scenarios are covered for the specific requirement

In traceability matrix we don't check if test case is having all possible scenarios

In test case review we don't check whether all the scenarios are for a test case is covered or not.

Q. How do you do traceability matrix without having requirement document?

I will go through application & I will list all the feature & I will check all the feature has got atleast 1 testcase or not.

## forward traceability matrix

SURYA Gold

Date \_\_\_\_\_

Page \_\_\_\_\_

Root document

Derived document

High level Req	LLR	Test case automation	Defect test script
----------------	-----	----------------------	--------------------

## backward traceability matrix

if both = bidirectional traceability matrix

mapping the requirement from root document to derived document it is called forward traceability matrix.

mapping the requirement from Derived document to root document is called backward/reverse traceability matrix.

And If It's functioning in both the direction then we call It as bi-directional traceability matrix.

## Test Execution:

This is the phase where we test the s/w more than 45-50 test cycle.

This is a phase where the tester are productive to the organisation & this is where we find the defect.

And this is the stage where we ensure that quality of product is good or not.

## Defect tracking:

While execution of test cases obviously we find a defect, those defects should be tracked in organised manner is nothing but defect tracking.

Anything which is not working according to customer's requirement is called defect.

We encounter defects because of the following reasons:

- ✓ ① Wrong Implementation
- ✓ ② Missing Implementation
- ✓ ③ Extra Implementation.

### Wrong Implementation:

e.g. Sent mail  $\xrightarrow{\text{X}}$  inbox  $\xrightarrow{\text{X}}$  trash

e.g. FB friend req sent  $\xrightarrow{\text{X}}$  after accepting automatically

e.g. A  $\xrightarrow{500}$  B (Amount transfer from A to B but happening from B to A)  
10,000  $\xleftarrow{500}$

### Missing requirement:

e.g. Gmail filter  $\xrightarrow{\text{Select all}}$  not there  
 $\xrightarrow{\text{X}}$   $\xrightarrow{\text{X}}$   $\xrightarrow{\text{X}}$

### Extra requirement:

If developer develop anything which is not ask by customer.

Q. what is the diff b/w defect, bug, error, failure?

Deviation from the req specification is called defect

Bug, It is the informal name given to defect.

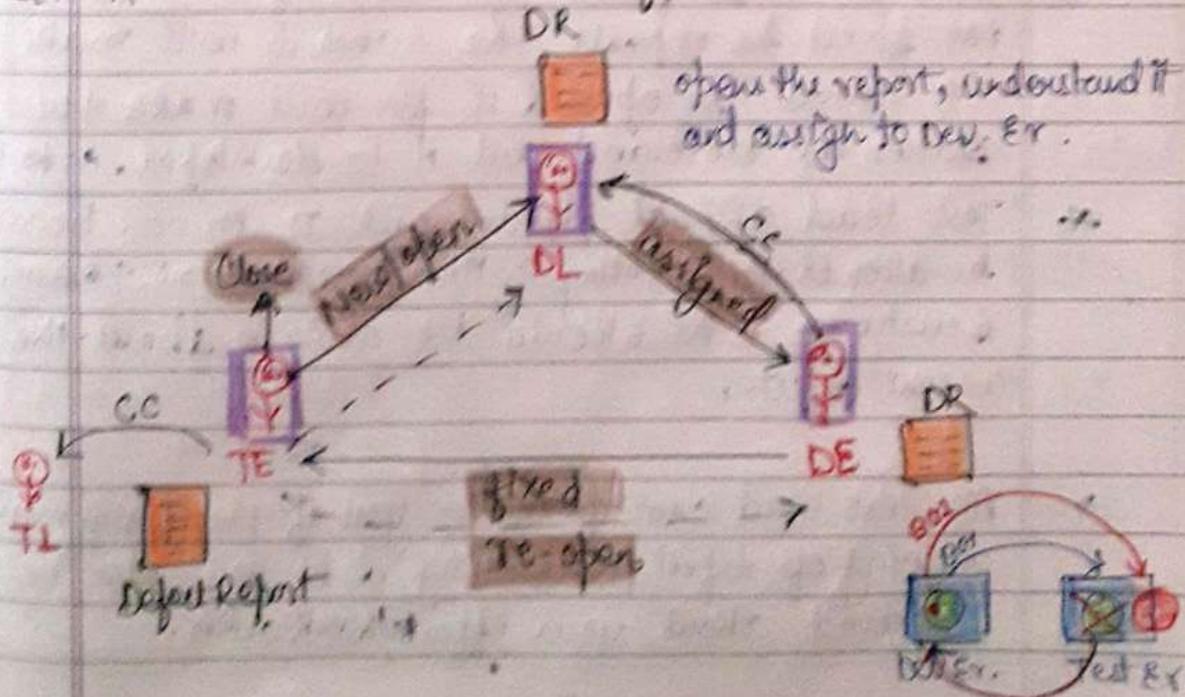
Error, It is the mistake done in the program because of that we are not able to compile or run the program.

There are 2 types of errors, one is compile time error and run-time error. Compile time error is because of syntactical mistake, run-time error is because of logical error.

Defect causes leads to failure.

error → formal (given by prog)  
 defect / bug → informal (given by TE)  
 failure → (given by customer).

Q. How do we track the defect?



- \* If the TE finds the defect, he will raise it by making a defect report & will sent it to Devlo (development lead if don't know the respective developer) or to Developer (directly if he knows who have developed that feature) keeping in sending cc to developer.
- \* While sending the defect report he will make the status as New/open.
- \* Now developer will open the mail & understand the defect, If its a really a defect he will sent it to respected developer to fix it & make the status assigned.
- \* Now developer will open the mail & will fix cross check it, if he find it as defect he will fix it & will create a new build & inform & sent to respective TE as well as put DL in cc & will make the status fix.
- \* Now if test Er. finds that the defect is still not fixed he repeats the same & will make the status re-open. & If fix will make the status as close and sent it to developer. etc to DL
- \* Test lead should always put TL in cc. because he attends the meeting with development team, PM & customer so he should be aware about the current defects.
- \* Also test lead can speak for you if the assignment or fixing of defect takes long. & he will also be aware about your efforts/work done.

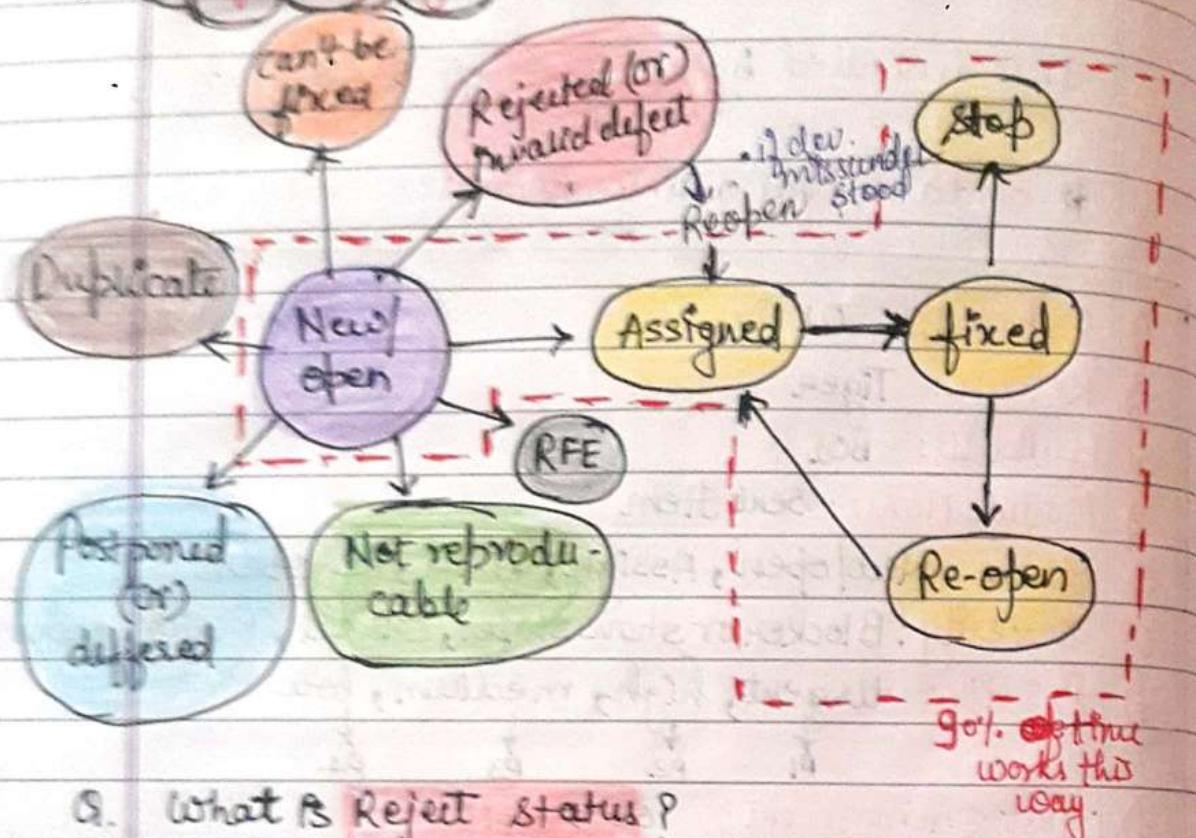
## How do we write defect report?

It is generated by using defect tracking tool.

It contains following attribute:

- 1) Defect ID: 01
- 2) Release: Tiger
- 3) Build ID: B01
- 4) Module Name: Sent Item
- 5) Status: New/open, Assign, Fixed, Re-open
- 6) Severity: Blocker or show stopper, critical, major, minor
- 7) Priority: Urgent, high, medium, low
  - P1
  - P2
  - P3
  - P4
- 8) Test Environment: OS → w10  
 Browser → Mozilla  
 Chrome  
 IE  
 Browser version →
- 9) Test data: Username: Ding, P/w: Dingi
- 10) Test case Name: Gmail-compose. All possible scenarios
- 11) Brief Description: Sent mail is not displayed in Sent Item page.
- 12) Detailed Description:
  - 1) Log in to gmail with valid login credentials
  - 2) Click on compose link
  - 3) Enter all the field & click on send button
  - 4) Click on SI link
- 13) Expected Result: Sent mail should be listed in SI page.
- 14) Actual Result: Sent mail is not listed in SI page
- 15) Attachment:  Browse.

## Defect life cycle:



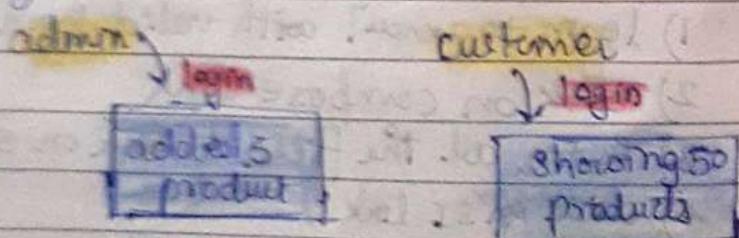
Q. What is Reject status?

Tester calls it as a bug & sent it to development team, development lead reject the defect & say it is a feature.

Q. Why we get reject status?

① Because of tester misunderstanding the requirement.

e.g. Amazon.



Searching for college bag  
It will show all sort of bags

Developer wrote such a logic to make impression to customer but TE might understand it as defect.

eg. In gmail click on save & cancel, mail will be saved to draft. It might understand it as defect.

② Test Err. might refer to old requirement.

eg. whatsapp <sup>old req</sup> group - only 1 admin

<sup>new req</sup> group - can have ~~one~~ multiple admin

③ Because of improper installation of the product

④ Because of extra feature.

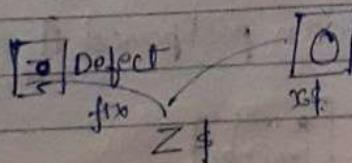
(anything which is not asked by the customer but added to the SWO it will be considered as defect only if)

⑤ Can't be fixed:

Developer, they are accepting it as a defect but they are not in the position to fix the defect because of some reasons.

① When ever technology is not supporting Dev team will say can't be fixed.

If the defect is critical they will have to change the technology & fix it. for minor & major they r not going to change the technology.



② When ever the cost of fixing the defect is more than the cost of defect developer might not fix it.

① new TE has come  
② common features

- ③ If there is minor defect in the root of the product developer says can't be fixed.

### Duplicate:

You sent a bug to development lead but the same bug is already sent by another TE so the DL change the status as duplicate.

Search the defect in defect repository before sending it to the development team.

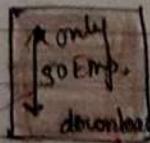
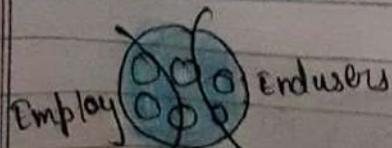
### Postponed

Development team, they are accepting it as defect but they are not in the position to fix it so they change the status to postponed.

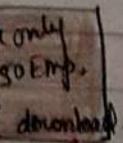
- ① If you find a minor defect & sent it to development team & DTR team is busy in fixing the critical & blocker defect, since they have insufficient time so they postpone the minor defect.  
eg. spelling mistake

- ② You found a bug in a feature when development team expecting a lot of changes from customer side, they make status to postponed

- ③ If there is a bug in a feature which is exposed to internal user, we can postpone it.



defect



alternati  
way

Excel sheet of with  
1000 Emp.

## Not Reproducible:

If you find a defect & send to developer he is not getting that defect.

- Because of Improper defect report -  
Developer is not able to reproduce the defect you have found, by reading your defect report.

e.g. gmail

↓  
login

↓  
compose  
↓

To  
cc  
Bcc  
⋮

**Send** **Cancel**

**Blank page**

You sent a defect to developer that clicking on to login then click on compose & send a mail, blank page is displayed

Developer opens up the report & try to do the same step but he couldn't find the defect you have found because you have not mentioned that on sending multiple mails we get the blank page.

- Same above e.g. for multiple attachments not getting send.

As Admin

e.g. create customer

Name [ ]

Phone [ ]

Address [ ]

Description [ ]

Active

**Send** **Cancel**

↑  
pop up msg  
this is  
above  
will not  
be able  
to select  
checkbox  
so screen  
shot also  
didn't help

You have sent the defect to developer that login as Admin fill all the entry & click on create an error msg display

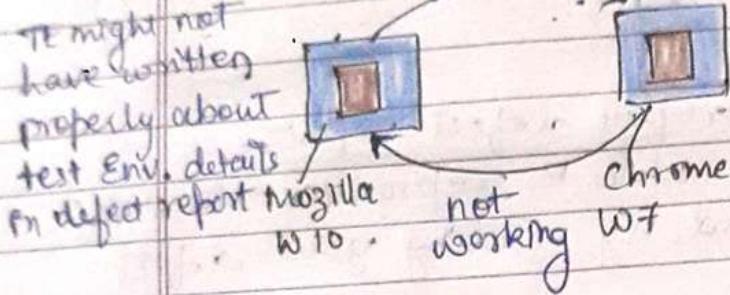
Dev. opens the report & tried it but he is not getting because defect was due to you have select the check box which is not to be selected & error msg was again pop up msg which was displayed above check box so screen shot also didn't help

### ② Improper platform:

TE might not have written properly about test Env. details in defect report.

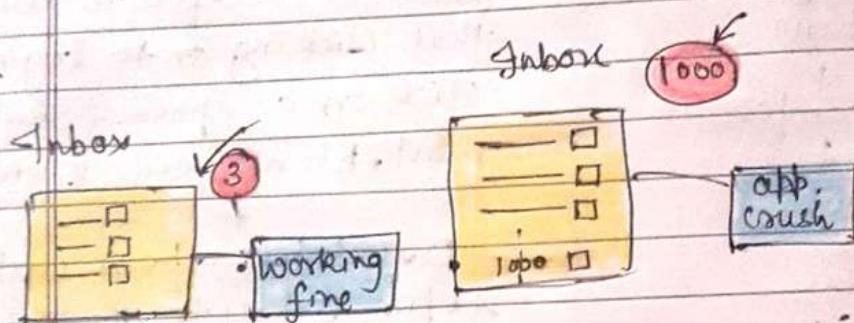
which browser?  
which OS?

He might not have read the Test Env. detail in defect report properly.



### ③ Because of Incorrect data.

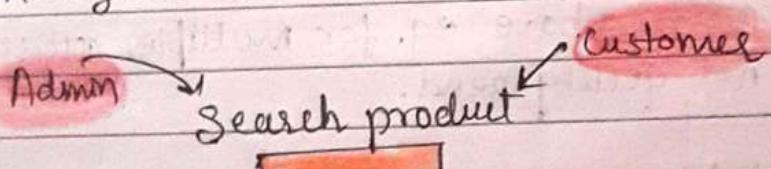
e.g1.



TE will sent the defect to dev. that inbox is crashing but for dev it's working fine. Here TE might not have realized that for 1000 mails its not working.

e.g2.

Amazon.

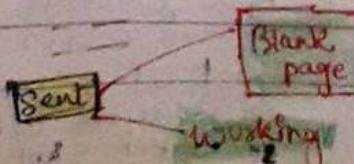


Here TE might have logged in as customer & search option for him is not working. But while rechecked developer might has logged in as admin & for him search option is working fine so he couldn't report the defect.

### ④

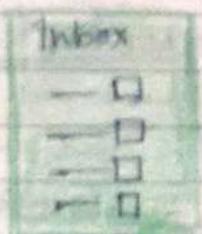
### Because of inconsistent defect.

Compose



## Request for enhancement (RFE)

eg.



Req: must select & delete 1 at a time.

TE: select all option is not there

eg.

whatsapp

Admin can  
only add  
& delete  
members

facebook

group

I will create 2 admin  
moderate feature (Who can post,  
add & accept req & send req).

TE: Add Moderate feature is not there

Defect Test Er. asking for any changes in  
the features it is called as Request.

for enhancement. Now PM will ask to customer,  
If he says yes than it will be added to  
that module by the developer.

### Defect seeding:

- Intentionally adding the defect in the S/O to check the efficiency of test Er. Is called Defect seeding.

### Defect masking:

- If I defect is masking / hiding the other defect Pt Ps called defect masking.

eg.

Approve OD  
(defect)

Approve OD  
(defect)

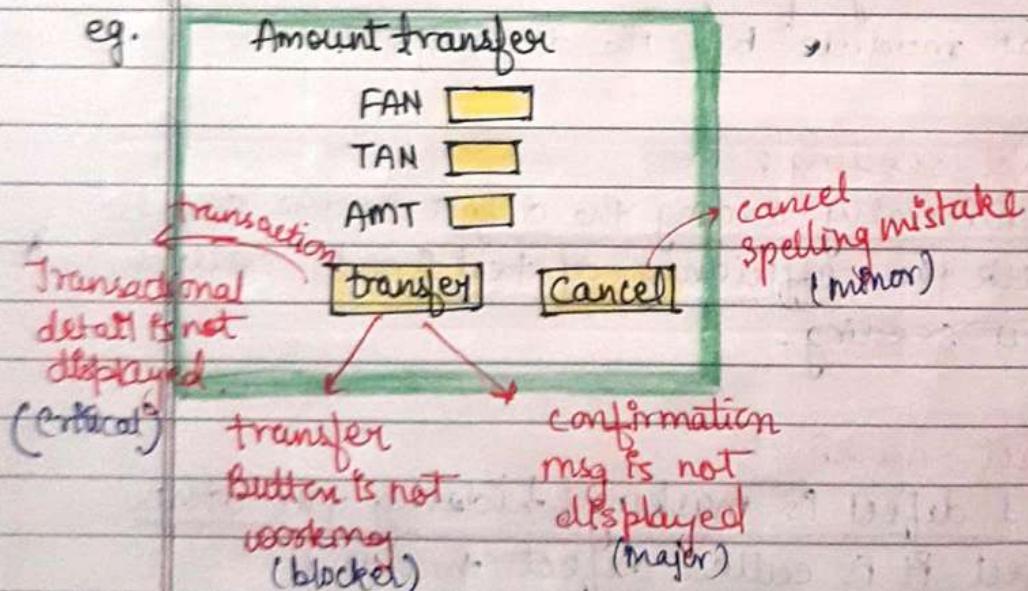
there is defect in approve  
approve OD thus this  
which can't be checked  
because there is defect in  
approve OD only.

- Sent mail button is not sending sending the mail  
than how can I check whether sent item that  
mail displayed in sent item list or not. If there is  
defect in sent item list.

## Severity & priority :

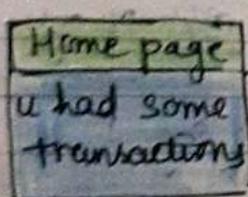
- severity is decided based on the impact of defect on ~~the~~ customer business.
- the severity levels are blocker or showstopper, critical, major & minor.
- which defect has to be fixed first is decided by priority or the order in which developer defect is fixed is decided by priority.
- priority levels are urgent, high, medium, low.

e.g.



e.g.,

login gmail,



95% of times

Blocker defects will have urgent priority

Critical will have High

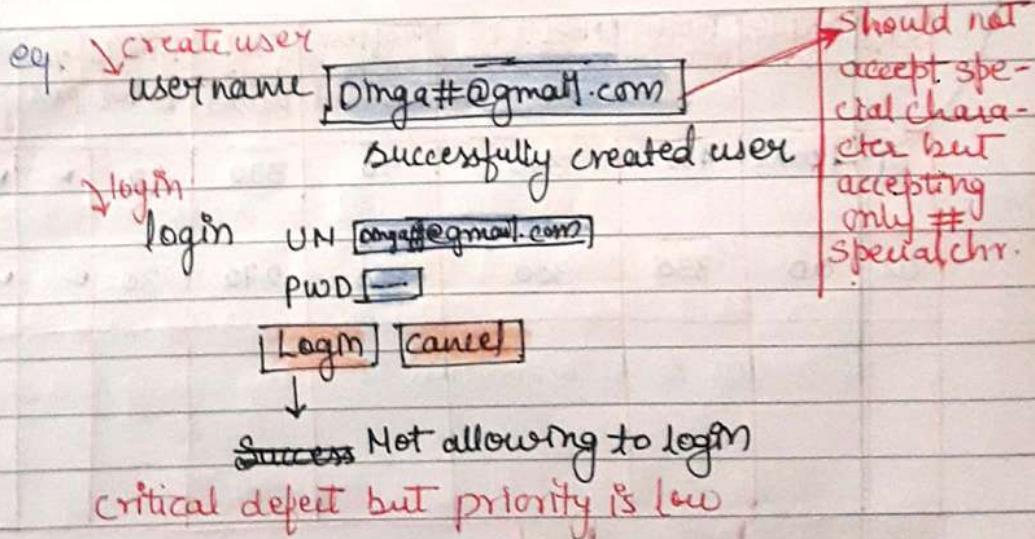
Major will have medium

Minor will have low

Q. Give an example for high severity & low priority

e.g. click on help link & if it take us to blank page, it is a critical defect but priority is low.

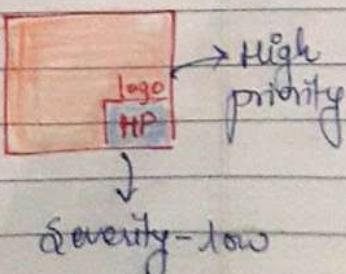
e.g. clicking on to calendar link & it is taking to bank page, it is a critical defect but priority is low.



Q. Give an example low severity (minor/major) & high priority?

e.g. Google → High priority

↓  
Spelling mistake  
(minor)



company logo & company name even when the severity is low but should be fixed first mean high priority.

## Test Execution Report:

loans

SNo	Test case Name	Status
01	Test case - H1	pass
02	Test case - approve loan	fail
:		
350		

loans\_approved

Step No	Action	I/P	ER	AR	Status

which step is failed you will know

SLNo	Module Name	Total test cases	No of test case executed	No of test case not executed	Total Test cases passed	Total TC %	Total TC %	% pass	% fail
01	Loans	400	350	50	330	20	10	✓	✗
02	OD	350	300	50	270	30	10	✓	✗

Summary | Loans | OD

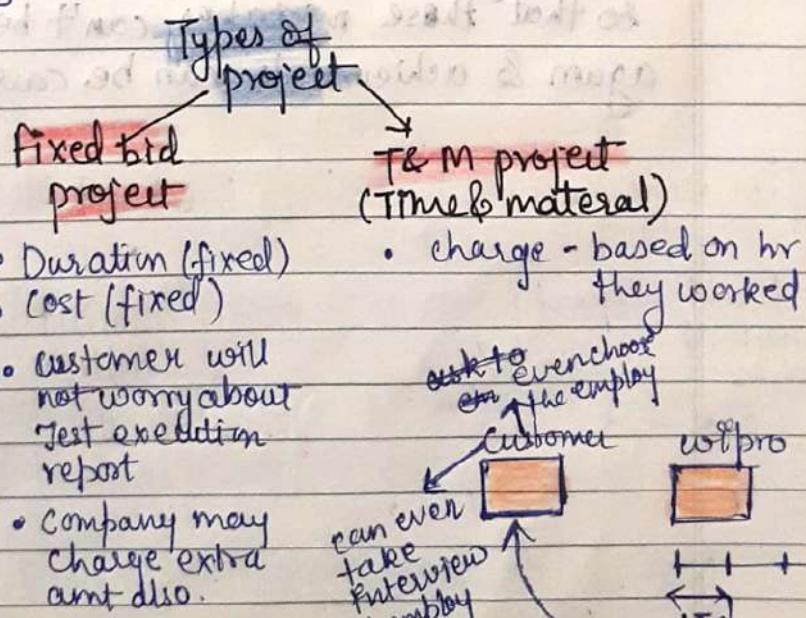
B01
B02
B03

At the end of every test cycle we prepare test execution report. It is a document which has got information about:

- ① Total Test case
- ② Total no. of Test case executed
- ③ Total no. of Test case pass
- ④ Total no. of Test cases failed
- ⑤ % of pass
- ⑥ % of fail

we send this doc. to dev team, business analyst & management. at the end of last test cycle we send the report to customer that is the end of test life cycle from the customer point of view.

from the testing point of view we have one more activity i.e. retrospective meeting or also called as proj. closer meeting or postmortem meeting.



In case of Time & material proj, customer can ask for ex-test execution report at the end of each test cycle.

### Retrospective meeting:

conducted by PM

Discuss → achievement?  
→ Mistake?

- we ~~may~~ have not reviewed the test case.
- we have done lot of full regression testing ~~and~~ instead of it, we would have done regional regression testing
- It is good that we have followed done traceability matrix.
- At the end of every test cycle we should have done Adhoc testing.

Achievements

Will document this & it is beneficial while writing the test case for next release/next so that these mistakes can't be repeated again & achievements can be carry forward.

## Adhoc Testing

also called as monkey/gorilla testing

Definition:

Testing the application randomly without following any referring any formal documents like requirement, test cases & test scenarios is called adhoc testing.

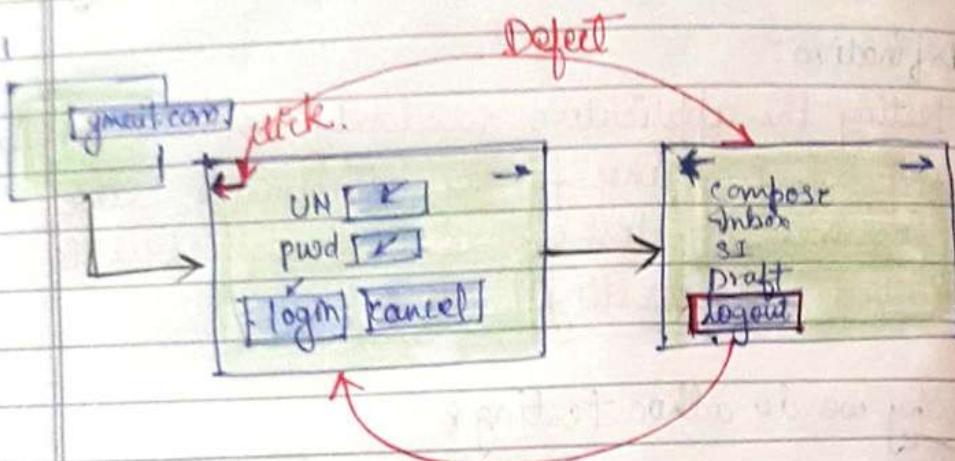
why we do adhoc testing ?

- ① When the product is launched to the production, chances are there end user might use the app. randomly & they might <sup>find</sup> some defect to avoid that test er. only should test the app. randomly & find the defect.
- ② If we follow the req & test the application the no. of defect we gonna catch is very less so we should come up with creative scenarios which is out of requirement & test the app. so that we can find more defects.
- ③ To somehow break the product we do adhoc testing.
- ④ To somehow ↑ the bug count we do adhoc testing
- ⑤ To check whether the s/w is working according to the implicit req. we do adhoc testing.
- ⑥ To somehow ↑ the test coverage we do adhoc testing.

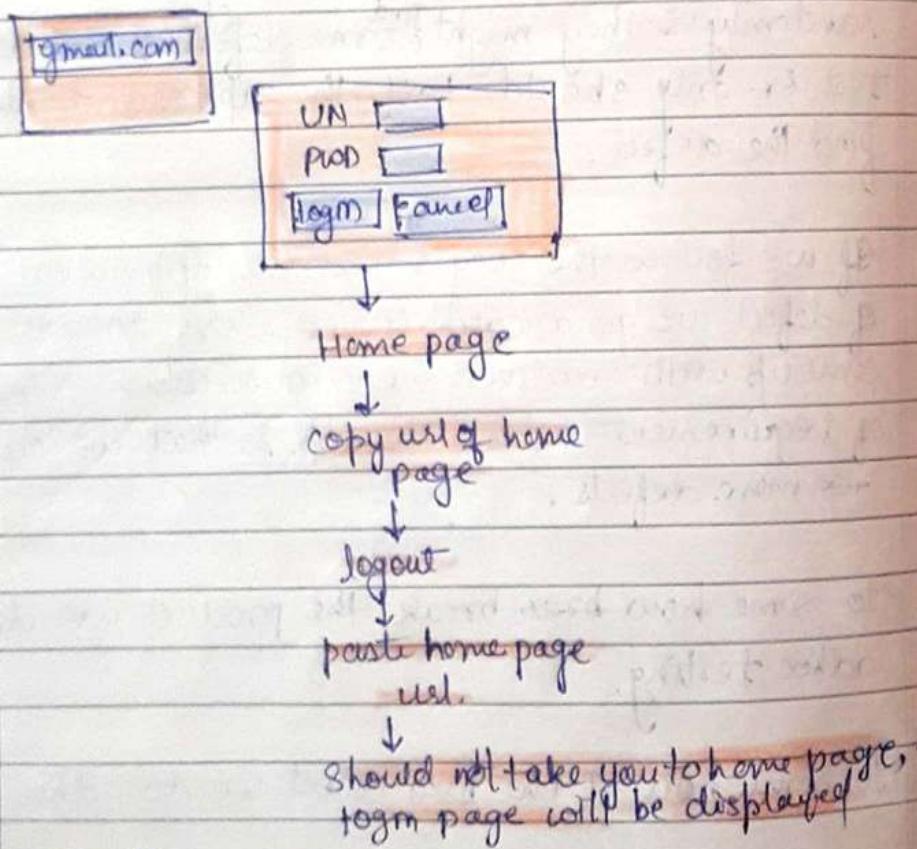
explicit req : given by customer  
implicit req: by common sense (e.g. customer give req for login it's should have logout this is common sense)

## How to do adhoc testing?

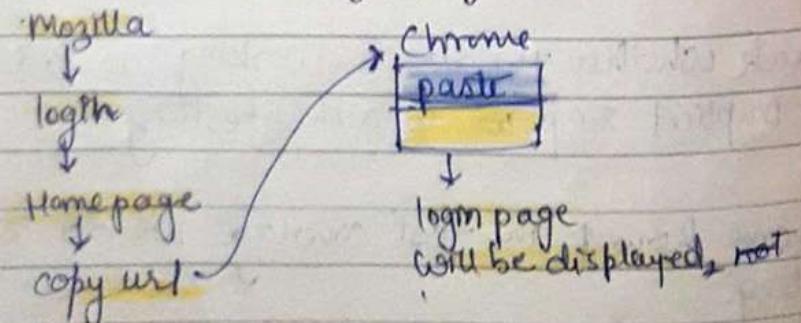
eg1

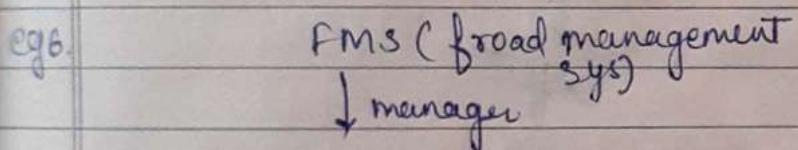
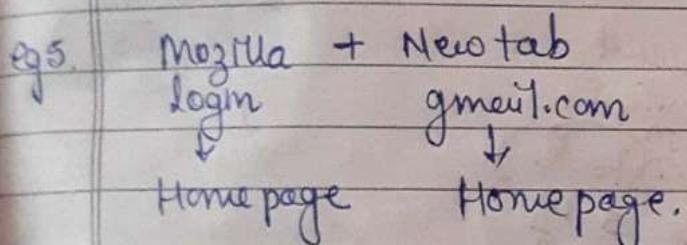
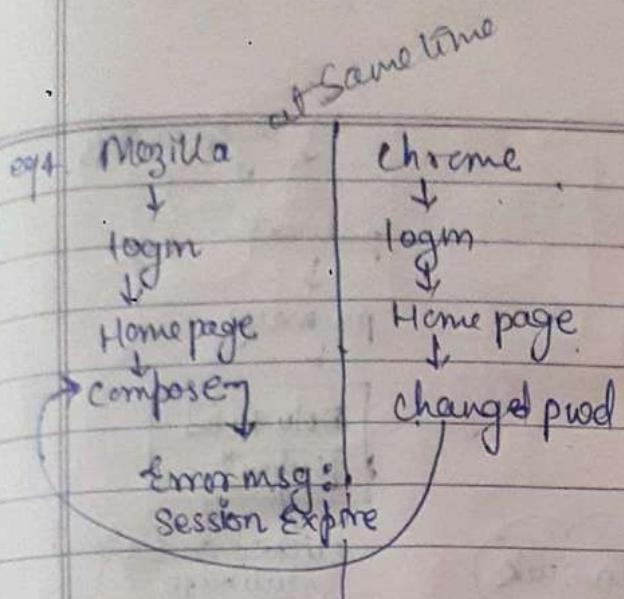


eg2. Mozilla



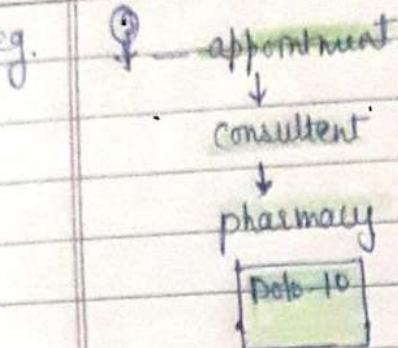
eg3.





- manager have blocked, & some body is transferring money to their blocked account it should not get transferred.
- loans feature might be processing for this blocked user.
- If this user is already login & Manager blocked it from that point of time only it should not work rather than - blocked user login

e.g.



→ ingredient diff  
size

patient

pharmacy

Data-to-10
Submit

clicked on submit  
submitting 10.

$$\Rightarrow \text{Stock} = \frac{15 - 10}{5}$$

Data-to-10
Submit

15 in stock

forgot  
to click  
on submit

again clicked  
on submit 1st

late on  
sys.

Stock

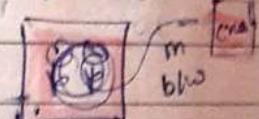
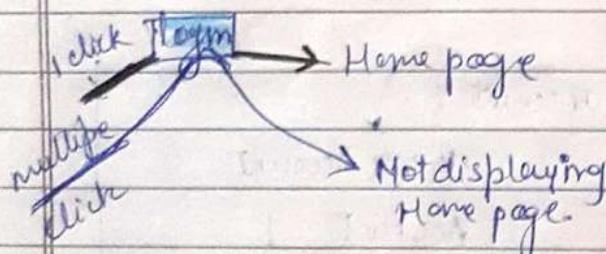
$$5 - 10 = -5$$

Defect

e.g.

U.S.   
pw

e.g. point



now they are not even allowing  
for multiple click.

e.g.

create user

user   
pwd

Submit  Cancel

→ Right click

copy  
edit  
cut  
undo

Blank

? It's a defect  
but user  
fix or not  
it depends

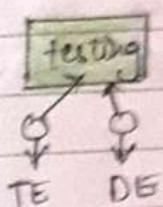
Q when we do adhoc testing?

Q when the application is relatively stable than we think of doing adhoc testing.

Note:

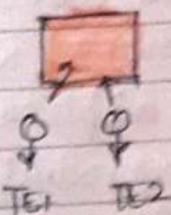
while doing smoke testing we should not do adhoc testing becoz smoke testing is +ve testing & adhoc testing is -ve testing.

pair testing



While doing pair testing & Buddy testing we can do adhoc testing.

Buddy testing

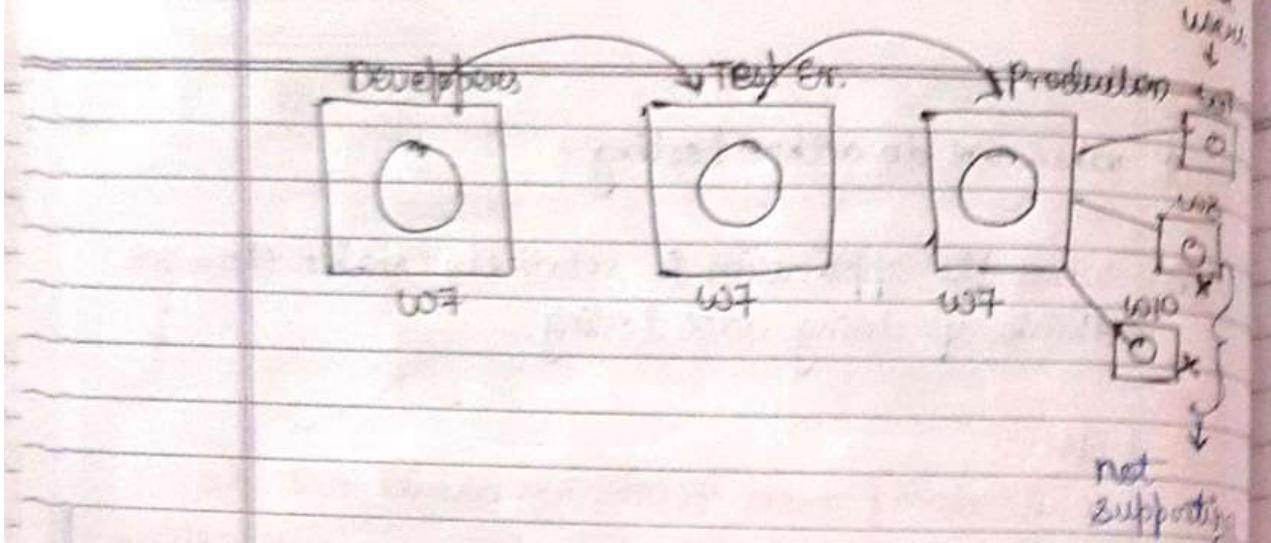


## Compatibility Testing :

- non functional testing

Definition:

Testing the application in diff. s/w & h/w platform is called as compatibility testing.

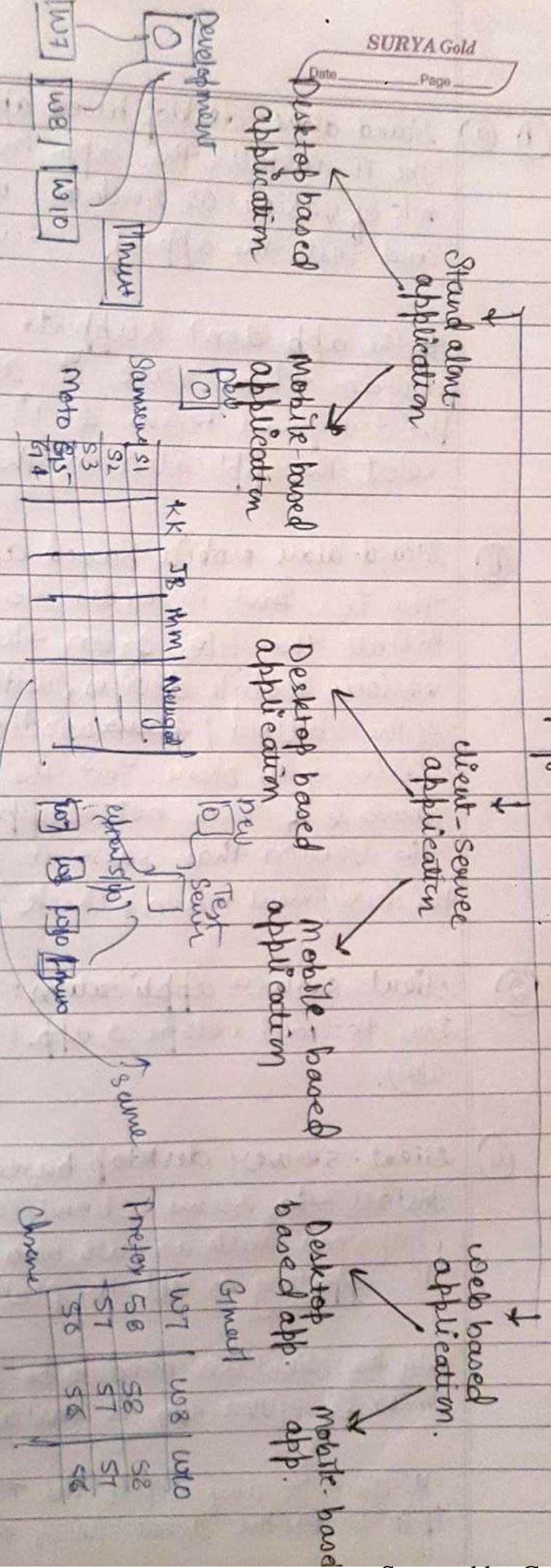


If the app. Is not tested in diff platforms  
It will not work for all the OS used by the  
end users Hence TE make sure they test the  
application in various platforms & if It is  
working fine then only they will launch it  
to end users.

### Why to do compatibility testing?

- ① So make so Developers develop the application in diff platforms for eg. windows 7, 8, 10 etc & if the TE is testing the application only in one platform & if the product is launched to end user then some of the features may not support in all the OS. In order to avoid this the application is tested in all the possible platforms or multiple platforms If not the business of the app will go down.
- ② Compatibility testing is done in order to provide **compatibility** to the app in all the platforms

**Projects**



① a) Stand-alone desktop based application:  
The TE installs the app. into various local m/c of various OS such as (W7, W8, W10 etc) and test the app if it supports in all the OS.

If the app. don't support in any of the platform then raise it as a defect to the development team & if they fix it retest the application in the same platform

b) Stand-alone mobile based application:

The TE's test in order to test the app. install the file from the server into various mobile devices, with various versions of the android / ~~Windows~~ iOS, & various series of the phone. Test the app. in all the above & if it is not supporting in any of the version than raise it as a defect to the development team & check if they fix it.

② client-server application:

Dev. team develops 2 app. (server s/w & client s/w).

a) client-server desktop based application:

Install the same client app. in various platforms such as W8, W10, W7 & test if it is supporting in all the platforms.

only the client s/w need to be installed multiple times & server s/w is installed 1's.

If there is any defect the TE need not find if it is in the client s/w or the server s/w.

Mobile-based client-server application:

The build given by the developer will be containing 2 s/w's of which server s/w will be installed in testing server & client s/w will be installed in diff mobile devices by the TEs & then they perform testing of that application. If they find any defect they will raise it to developer & this cycle will continue until s/w become stable. Once s/w will be stable, it will be given to customer, & <sup>server should be</sup> installed in production server & client s/w will be installed in some portal like appstore, playstore, <sup>play from where the customer</sup> can download & install. In order to test the app, install the s/w file into various mobile devices.

Web based application:

Desktop based

⑥ Mobile Based.

Install the same app from various platforms such as web, w10, or test if it is supporting all the platform.

Install the s/w into various mob devices, with various versions of android / ios

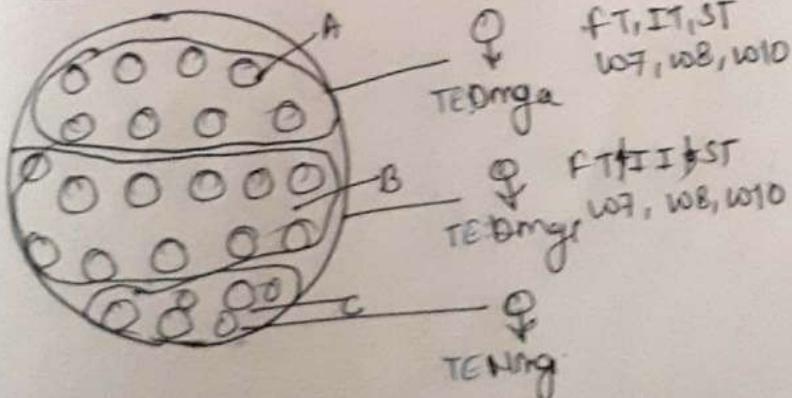
& various series of the app.

Test the app in all the above & if it is not supporting any of the version then raise it as a defect to development team.

Note:- Compatibility testing is not a separate testing. It is performing FT/IT/ST in diff platforms.

Q How to allocate work in compatibility testing?

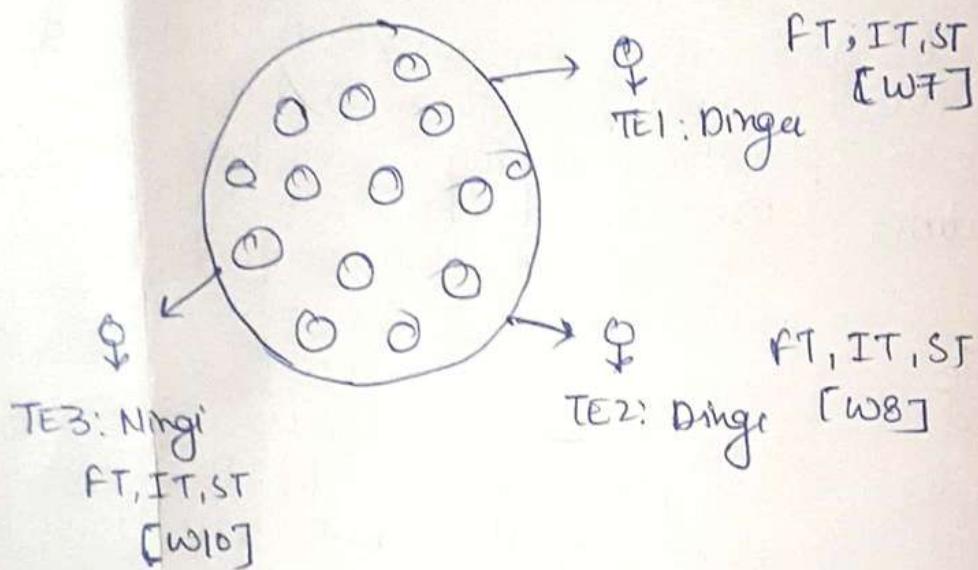
1st Approach



Compatibility testing → Performing FT, IT, ST in diff platforms.

group & segregate all the related modules & features & assign it to TE1, TE2 & TE3 & ask each TE to perform FT, IT, ST on different platforms.  
 e.g. TE1 can assign all the modules of A & the performing FT/IT/ST on windows 7, 8 & 10.

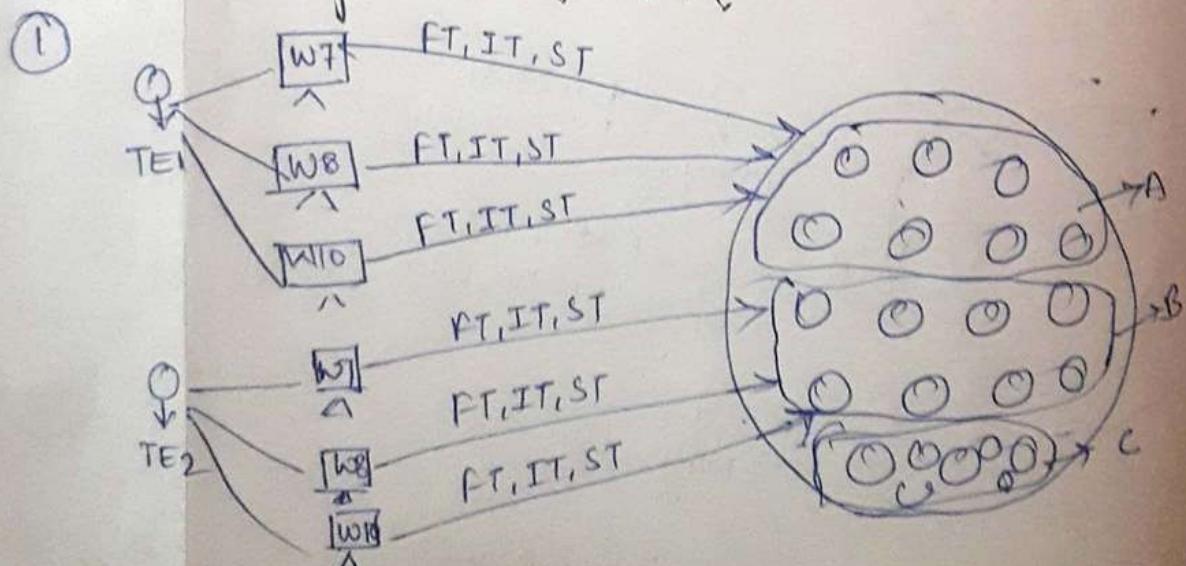
## 2nd Approach



All the modules of the application is assigned to 1 test where he performs FT, IT, ST in only 1 platform. Similarly another TE performs FT, IT, ST on all the modules in one platform.

This is not a good approach & is normally not followed.

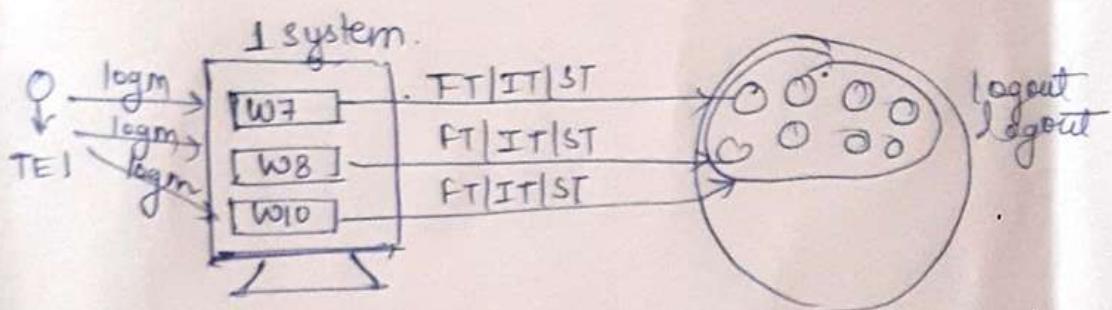
How to setup an Env. for CT?



Here each tester will be allotted three diff m/c with 3 diff OS & then they will test the module allotted to him.

If the app is not running well on any of the SW he will raise it as a defect that this app. is not supporting that particular OS. But in testing way we need more no. of systems

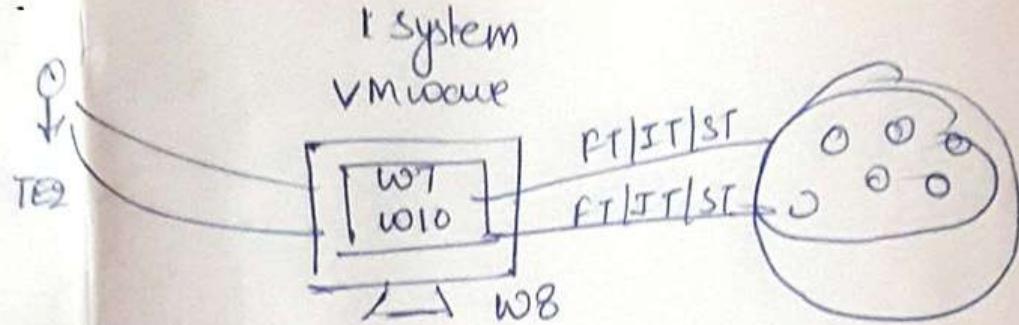
2).



Here TE1 will be given only a single sys installed with all the 3 diff OS in it & he will have to log in with for eg. W7 & check the app. & logout & similarly check for another OS also.

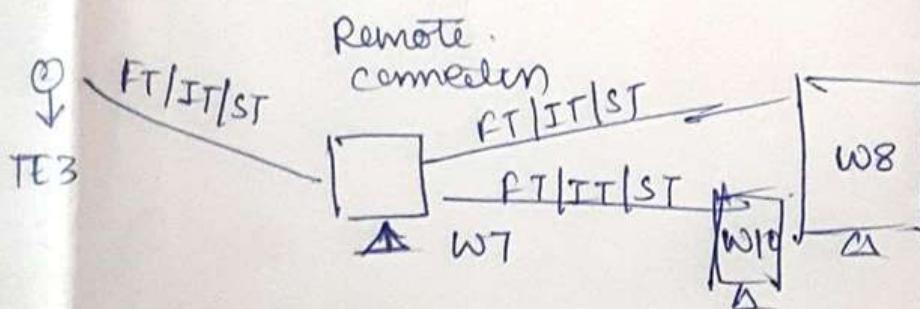
If he finds the defect that if the app doesn't run on any of the OS he will raise it as a defect that that particular OS is not supporting. But in this approach performance issue will be there so its not better approach.

3).



Here ~~some~~ diff OS will be Here we will be having only 1 sys in which a virtual m/c is installed which will allow to access that sys with W7 & W10 also & then the TE will be testing in all the os.

4).



Here we can make remote connection with sys diff containing diff OS & check ~~if~~ on the apps compatibility in all the OS.

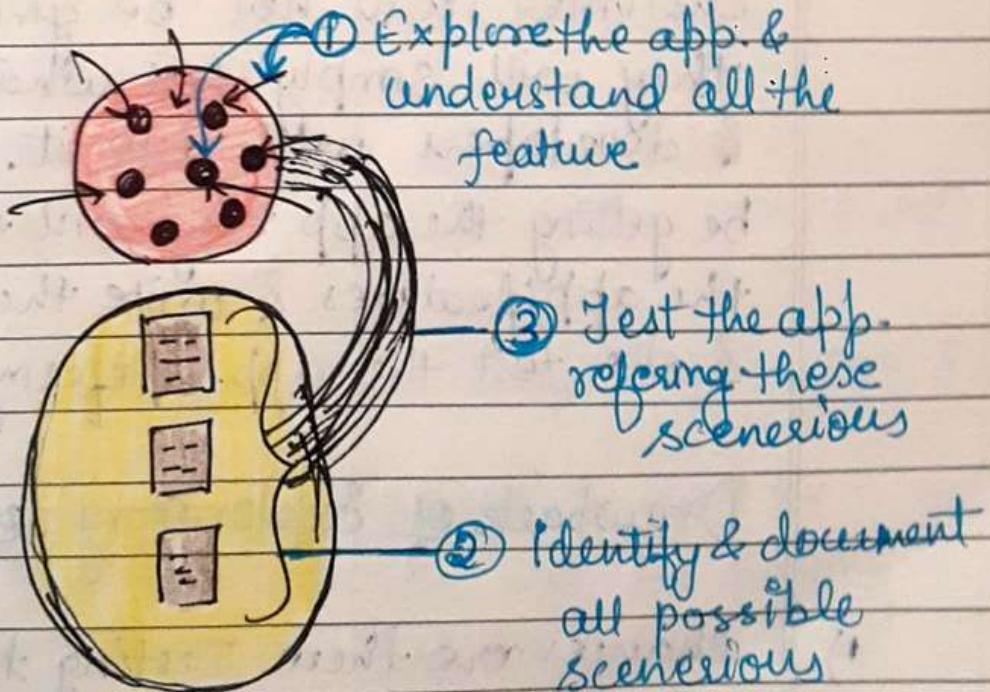
# Exploratory Testing:

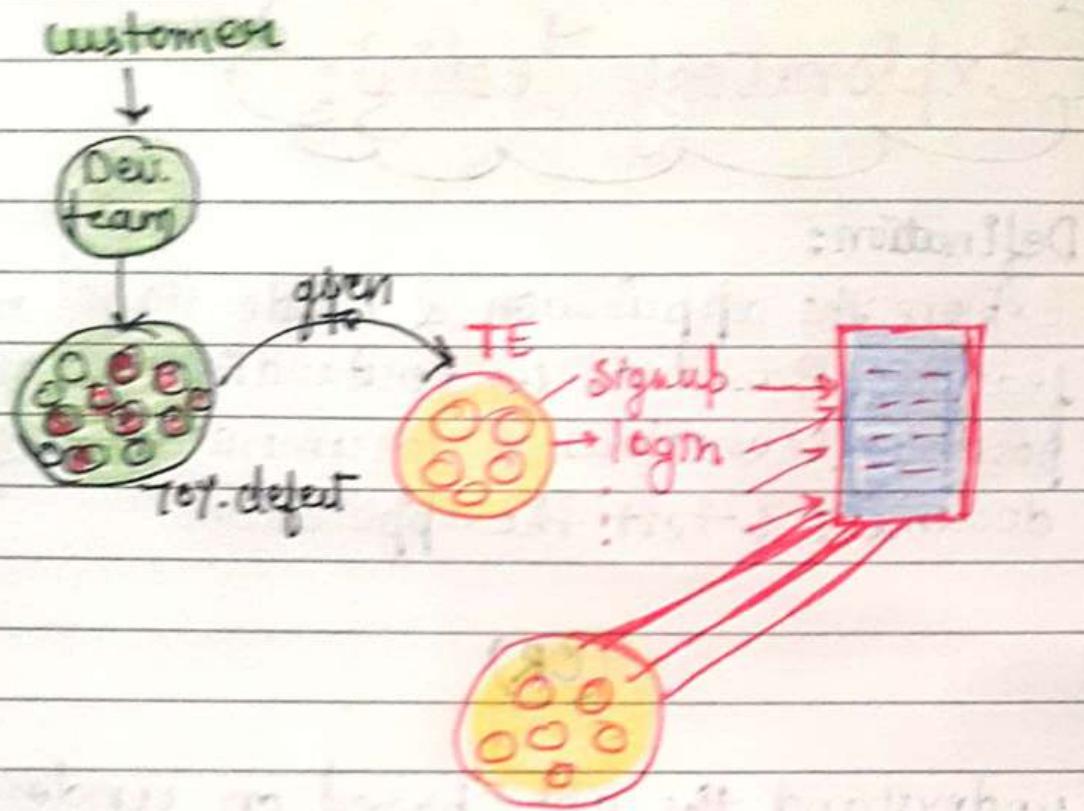
Definition:

explore the application & understand all the features. Based on understanding identify all possible scenarios & document it. Refer the document & test the application.

(OR)

understand the app. based on understanding identify scenarios & test the application.





customer will not be giving any CRs  
 they will simply tell what they want  
 & developers will do it. Now TE's will  
 be getting the app they will explore & understand  
 the app features & write the possible scenarios  
 & test the app. referring these scenarios.

### Drawback of Exploratory testing:

- 1). Chances are there Testing team might misunderstand feature as a bug & defect as a feature.

If req. change or new req. comes & TE is not aware  
he will be checking with old understanding only.  
**papergrid**

Date: / /

- 2) If the feature~~ing~~ is missing we will never come to know that it is missed.

How do you work on the above problem?

- 1) By interacting with developer, BA & customer very closely.
- 2) Based on the domain knowledge - try to test the application.
- 3) Based on the knowledge of the proj, which is already released to the production.
- 4) When ever there is no req. we can take error msg as a req. & test the application. (IT not tested)

How do they manage exploratory testing proj?

The TE what ever he understand he will be documenting it & here also we will be having test plan & test cases so that in future if he quits the job, other TE can perform the testing referring those TP & TC.

## When or why we do exploratory testing?

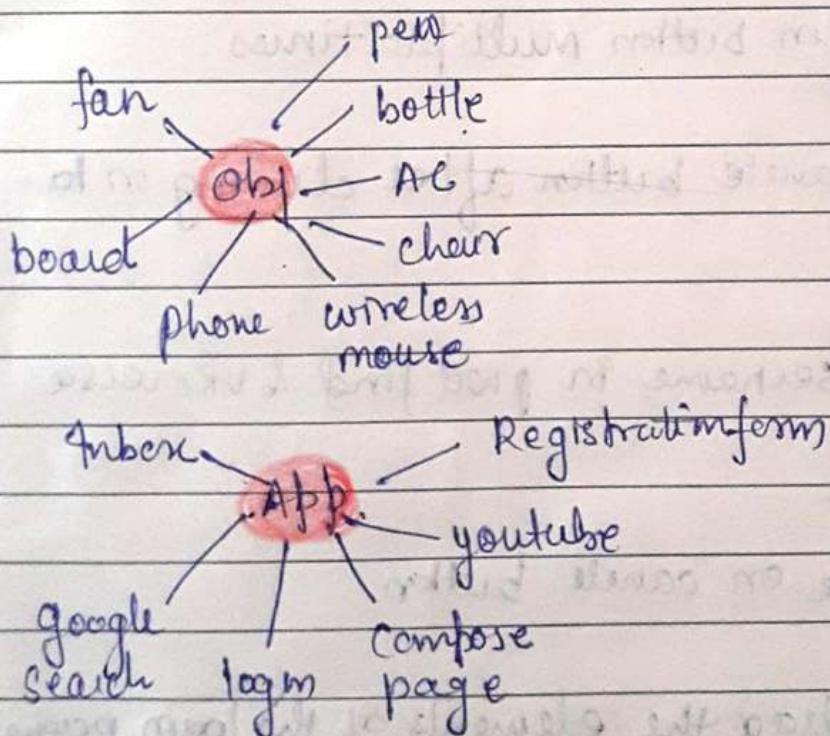
- 1). When ever there is no requirement.
- 2). When ever there is a requirement but not understandable.
- 3) Req is there & its understandable but the time is not there to go through it.

<del>what</del>	Smoke	Adhoc	Exploratory
when?	Testing the basic or critical features of an app without doing before doing thorough testing.	Testing the app randomly without any formal doc. like Req, TC & FP Test scenarios.	understanding the app. based on understanding write the possible scenarios & test the app going through these scenarios.
why?	When ever we get the new build	When the app is relatively stable	When we don't have req.
	To check whether the build is testable	To ↑ the bug count	Req is not there

+ve or -Pre	positive	negative	both
Req.	we have refer	we don't have refer	we don't have.

→ obj application.

How do you test - - - ?



ST, FT, JT, ST, Adhoc, compatibility, usability, acceptability, recovery, globalization, performance.

# Globalisation Testing :

Definition:

Developing the s/w for multiple languages is called globalisation.

Testing the s/w which is developed for multiple language is called globalisation testing.

There are 2 types of globalisation testing:

- 1). Internationalization [I18N] testing.
- 2) Localization [L10N] testing.

Any s/w developed for multiple lang will have 2 sort of files one is properties file & other is program file.

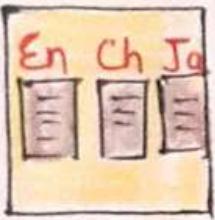
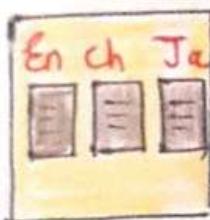
Previously translator tool were used but it don't work fine e.g. welcome → Hindi

↓      ↓  
ଓ ৰ ৰ    ম ৰ ৰ

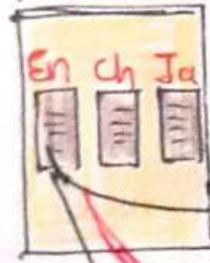
# SERVER

D: Properties file

login.prop sales.prop

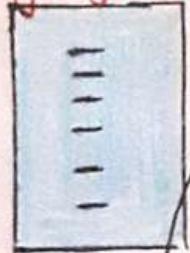


welcome.prop forum.prop

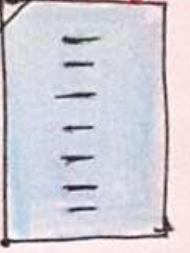


D:\Programfiles

login.java



sales.java



Request object

- ① page name: welcome
- ② Lang code: en

contain content only.

en.properties

ch.properties

Link1 → login  
Link2 → forum  
Link3 → Sales  
Link4 → Logout

Link1 → ??  
Link2 → ??  
Link3 → ??  
Link4 → ??

pull this

display user

URL www.google.com

Lang	English
	Chinese
	Japanese
	...

## Internationalisation (I18N) testing:

Here we check whether the right content is displayed in right language or not.  
& right content is displayed in right place or not.

we do this by adding suffix & prefix to properties file & it is also called ~~pseudo~~  
~~correct~~  
~~content~~  
~~prefix~~  
~~suffix~~  
~~pseudo~~  
pseudo translation.

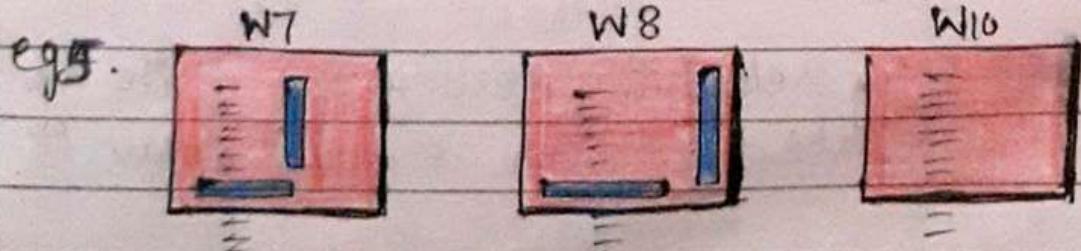
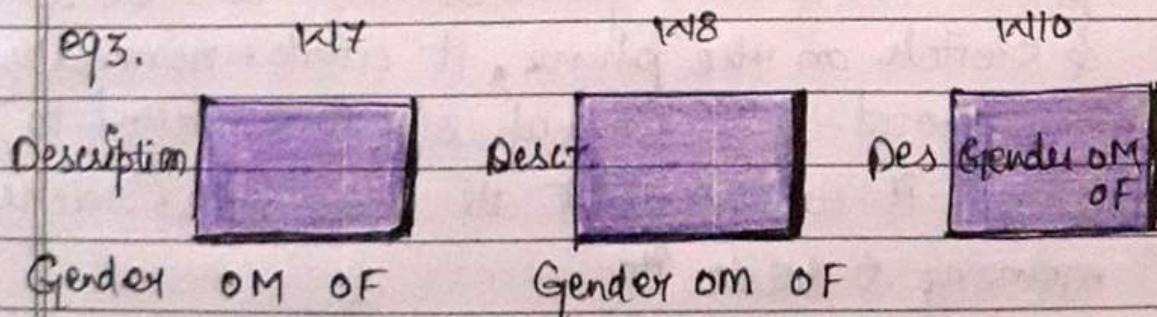
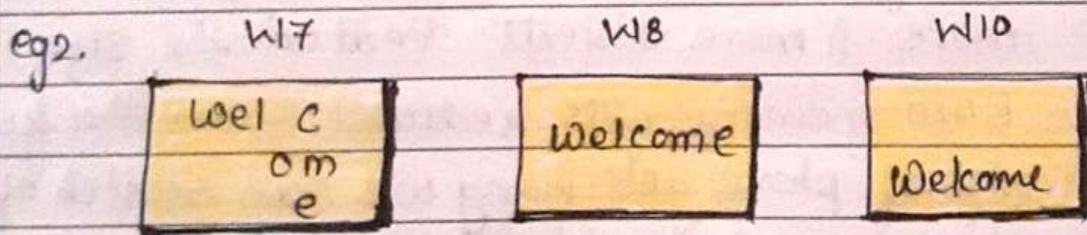
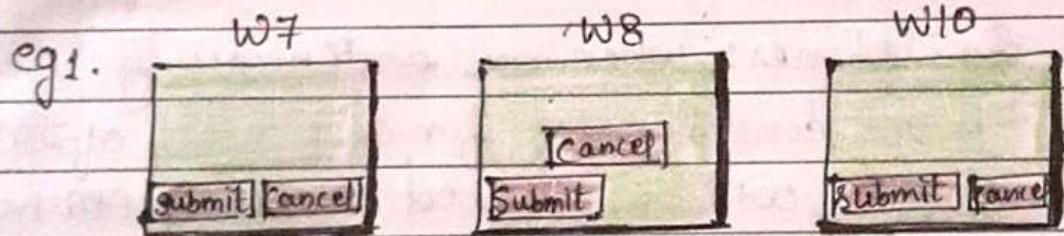
## Localisation testing:

Check whether the features are changing locally according to country standard or culture is called localisation testing.

e.g. time, currency,

# Compatibility testing Defects

1. Alignment issue
2. scattered objects eg: Blurred images
3. object overlapping
4. change in look & feel (color, font, size)
5. scroll bar display issue.



# Reliability Testing

→ Non-functional testing.

Definition:

Testing the applications functionality continuously for a particular period of time.

Eg: phones: when we continuously use the phone continuously for a period of time, the objects will be created in the RAM memory every time when we use the phone. When more & more objects created the size of the RAM memory gets reduced. Hence the functionality of phone gets hang we can switch off the phone so that objects created will be cleared & switch on the phone, it works normally.

To avoid this cleanup s/w is installed in phone It will collect all junk files in cache memory & delete it.

For doing reliability testing we write an automated program or script & run it.

In order to do reliability testing we use automation tools like WEIBULL++, RGA tool (Reliability growth analysis)

## Recovery Testing

→ Non functional testing.

Definition:

Testing the application how fast it recovers from a crash or disaster is called as recovery testing.

only senior Test Engineers or trained test engineer perform recovery testing.

steps to do recovery testing:

1. Introduce a defect & crash the application
2. whenever the s/w crashes, it should display a error log/ crash log msg mentioning the reason for the crash.
3. It should kill its all process before it disappears.  
eg. Task manager which shows all the process running in the background wherein end user

can choose the process & end it.

- Re-open the application wherein all the previous settings should be restored.

e.g. closing the laptop abruptly wherein chrome browser open with 8 sessions & later once we restart the laptop again go to chrome browser it should display a crash log, start a new session.

e.g. Network Issues: Issues wherein from high ~~n/w~~ n/w area when a download process is happening & immediately travelling to low ~~n/w~~ n/w area which will pause the downloading & once again we return to high n/w area. It should start from the place where it was paused & continue the download the rest of the file.

If it starts from initial place again proper recovery management mechanism is not updated. Hence loss of data, loss of money, loss of time.

# Accessibility Testing / ADA testing / 508 act testing

→ Non-functional testing.

Definition:

"Testing the application from a physically challenged person's point of view is called as accessibility testing".

ADA - American Disability Act.

It is a Non-mandatory testing. If the application has to be released to multiple disabled users then we do accessibility testing.

e.g. Google maps.

We can't do accessibility testing manually by a normal tester. Hence we use automation tools like.

- 1). Wave
- 2). Infocus

## Agile :

It is a iterative & incremental approach. The main goal of agile is customer satisfaction by quick delivery of working software.

## Principle of Agile methodology :

- Customer can change the requirement at any point of development stage.
- Release should be short
- There will be good communication b/w customer, BA, developer & Test engineer
- It is simple model to adopt
- Developers & test engineers will be doing lot of meetings at regular intervals in order to improve the quality of process.

## What is agile testing?

Testing the software by following the principle of agile methodology is called agile testing.

## Types of agile methodologies?

1. Scrum methodologies
2. XP (Extreme Programming)
3. FDD (feature driven development)
4. Crystal clear.
5. Lean & kanban
6. ASDM (Adaptive software development method)
7. DSDM (Dynamic software development method).

## Scrum methodologies | Scrum Process:

1. Release: Combination of sprints is called release.
2. EPIC: Complete set of requirement is called EPIC. One EPIC consist of multiple features or modules.

## User stories/stories/story cards:

Features or modules of functionalities is called as stories one EPIC consist of multiple stories.

## Story Point:

It is rough estimation given by developer & test engineer to develop & test individual stories.

Ex: 1 story point = 9 hours

1 SP = 6

1 SP = 8

Story point should be in Fibonacci series.

## QA Story points estimated based on the following aspects:

Time spent on understanding story

Write test scenarios

write test cases

Review test cases

Test case execution

Defect tracking.

## Developer story points estimated based on the following aspects:

Understanding requirements

Design

Coding

Reviewing code

WBT

Time spent on fixing the defects.

**Swag:**

The rough estimation given by developer & test engineer for every individual stories in the form of hours.

**Sprint:**

Sprint is the actual time spent by developers & test engineer to develop & test one or more stories.

**sprint planning:**

Sprint planning is a meeting conducted by scrum master on the first day every sprint. Explain the requirement, identify list of task to be done, Assign task to engineer.

**Scrum Master:**

Scrum master is a person who is responsible for delivery of the software to the customer within a planned period of time. Scrum master will track all the activities done by

developer, test engineer & BA.

BA/ senior test engineer/ senior developer/ project manager & even customer can become Scrum Master.

### Scrum meeting:

It is a meeting conducted by a scrum master on the daily basis.

It is also called as **daily stand up meeting**. This meeting is strictly bounded for 15 to 20 minutes.

In this meeting we discuss below mentioned points

1. What you did yesterday.
2. What have you planned today
3. Are there any obstacles or impediments

### Sprint retrospective meeting:

It is a meeting conducted by scrum master on the last day of every sprint. In this meeting we discuss the following points.

1. what went well
2. what didn't go well
3. Are there any actions plan

### Release retrospective meeting:

It is the meeting conducted by scrum master on the last day of every sprint.

### Bug trace meeting:

This is the meeting conducted by the test engineer or Scrum master a week or 2 week before the release BA, TE, Developer & SM will be a part of meeting

In this meeting test engineer list all the open & pending bugs which are not fixed by the developers in the current & previous release as a team we will re-prioritize the defect from the customer's business point of view & decide how many bugs should be fixed as part of a current release & how many bugs can be move to upcoming release.

## Product Backlog meeting:

This is a meeting conducted by a BA or SM a week or 2 week before the release.

In this meeting test engineer & developer come up with list of all the pending stories

which are not implemented as part of the current & previous release.

As a team we will re-prioritize all the stories from the business point of view & move the stories to the next upcoming sprint or release.

## Resume points:

### Experience:

Worked on Agile Scrum Development process.

Involved in Sprint planning meeting, Daily Standup meeting & Sprint Retrospective meeting.

### Fresher:

Excellent knowledge on Agile Scrum Development process.

Very good knowledge on sprint planning meeting.

Very good knowledge on Scrum meeting.

Very good knowledge on sprint Retrospective meeting.



# ALM / QC

papergrid

Date: / /

ALM: application life cycle management tool  
QC: Quality Center.

- \* It is a test management tool, used to manage, testing activities of the project.
  - ↓  
entire STLC

- \* Testing activities of the project involves
  - ① Requirement management
  - ② Test Design
  - ③ Test coverage
  - ④ Test execution
  - ⑤ Defect management
  - ⑥ Report management.

## \* History:

Mercury Interactive Cooperation — Test Director — 1985 — 1.0

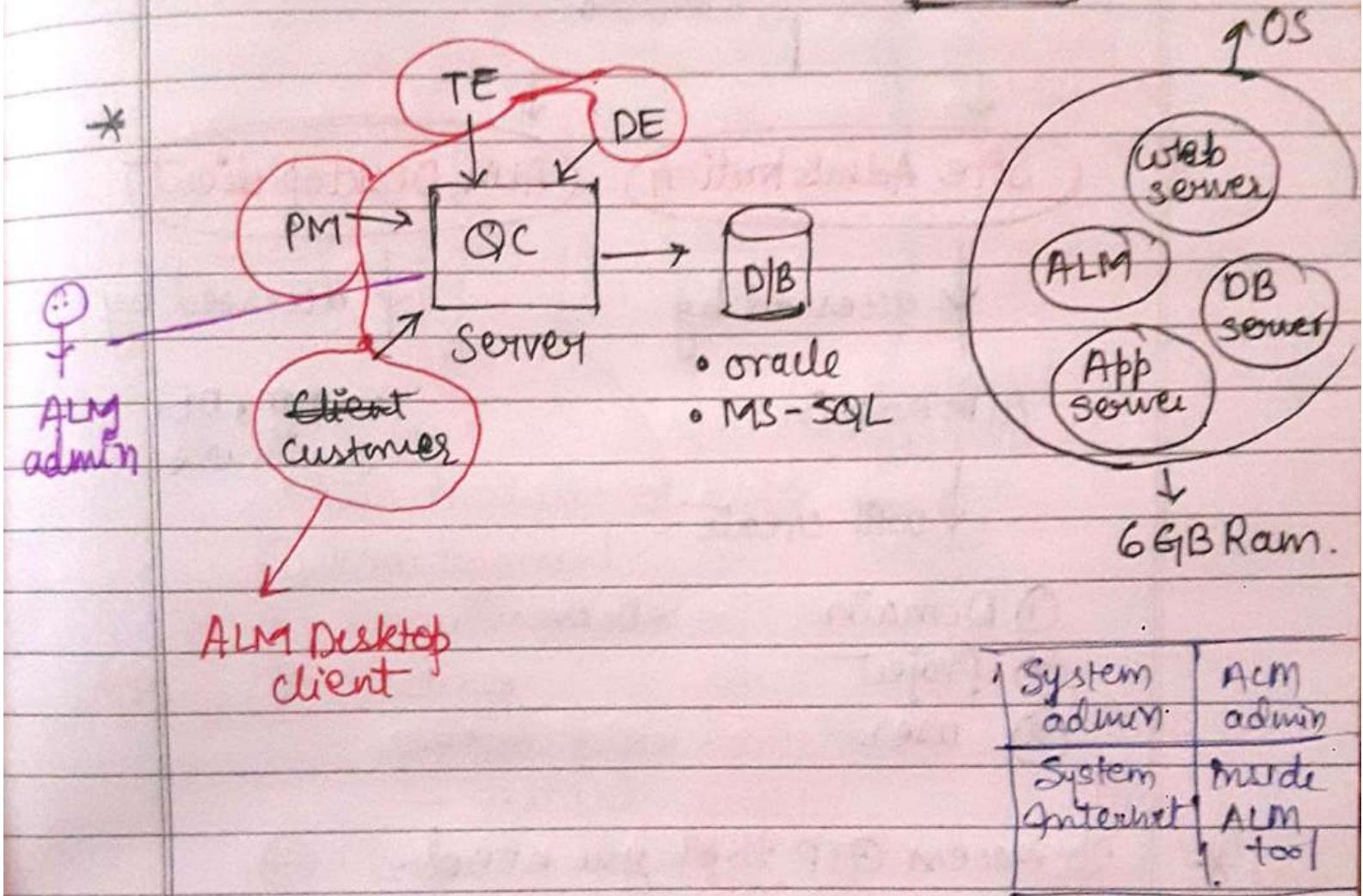
- " — QC — 2004 — 8.5

HP — HP purchased — 2007  
& renamed  
as HPQC

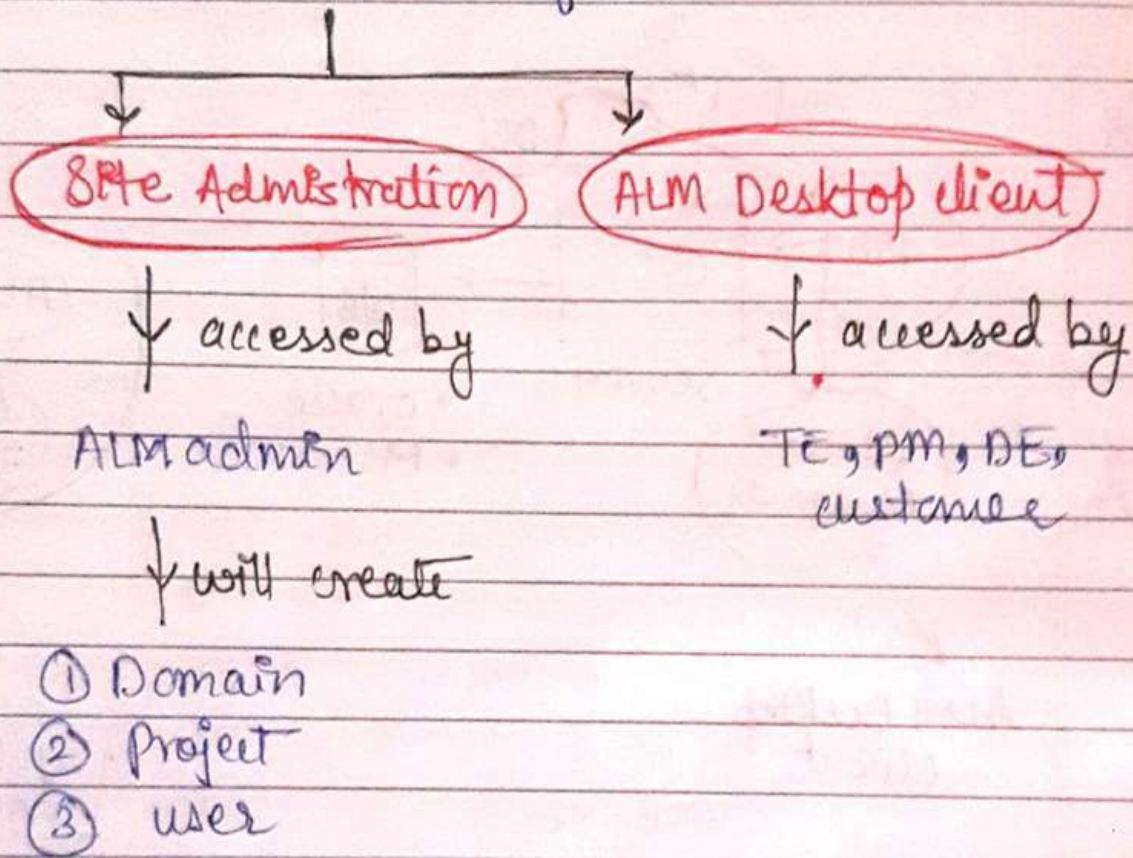
- " — ALM — 2011 — 11.5

current latest version -

12.53



\* QC tool has 2 Interface

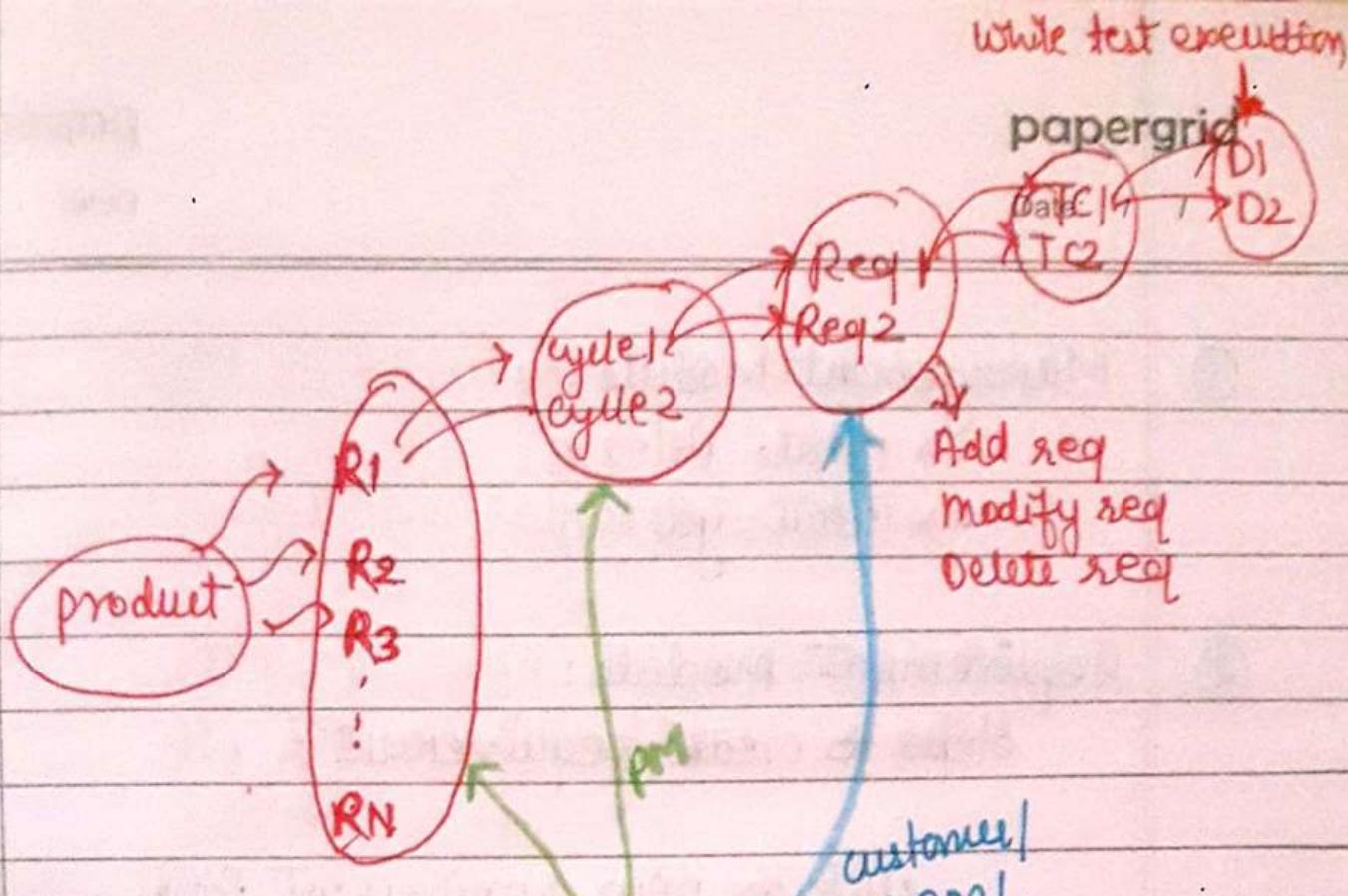


\* To access QTP tool you need.

→ URL : https://Server name:8080/QC bin

↓                  ↓  
IP Address post No.

- UN
- Pwd
- Domain name
- Project



Main modules of ALM

- ① Management
- ② Requirement
- ③ Testing

→ Test plan - write TC  
→ Test Lab - used for

- ④ Defect

test execution,  
store copy  
of TC to be  
executed in  
each build.

customer/  
BAI PMI  
you can  
read only

In XL



export but  
same template

- If so build → so execution results.

B01 → B02

Defect  
(B01)

checks defect (D1)  
in ~~the~~ next build.

(Raised defect) (Recheck defect)

## ① Management Module :

- create release
- Create cycle.

## ② Requirement module :

Steps to create requirement:

Click on new requirement icon



enter the requirement name  
& requirement details



click on submit button.

- you can download the requirement in your local machine.

Ques. How do you download requirement document from ALM?

Go to requirement module



Open the requirement



Click on attachments option



Select the requirement document



Click on save as button, enter  
the file name & click on save  
button.

### ③ Testing module:

#### → Test plan module:

It is used to write test case as well  
as automation test script.

Steps to write test case:

Go to test plan module



Select the requirement  
test case folder (1st we  
should create)



Click on new test icon,  
& enter the test case name  
& objective of test case

Inside description



Click on Ok button.



Click on new step icon,  
enter the description  
& expected result



Click on Ok button



Repeat the above  
steps (last 2 steps) for  
all possible scenarios.

### Traceability matrix:

case 1: Backward traceability matrix ( $T_C \rightarrow R_{req}$ )

Go to requirement module



Open the requirement



Click on test coverage  
icon.



click on select icon



select the required  
test case on on the  
test plan tree &



click on left pointing  
arrow mark.

case 2: Forward traceability matrix (Req → TC)

Go to test plan module



select the test case



click on select requirement  
coverage



click on add to coverage  
icon (left pointing arrow).

### → Test lab:

The module is used to execute manual test cases as well as automation test scripts.

#### Steps to create test case:

##### ① Steps to create test set folder:

go to test lab module



click on new test  
set icon



enter the test set  
name & click on  
OK button.

##### ② Steps to add test cases to execution grid

click on selected test from



Select the test case  
in the test plan tree

↓  
click on left pointing  
arrow mark

### ③ Steps to execute test cases

Select the test case, in the  
execution grid

↓  
click on run button

↓  
click on begin run button

↓  
see the description &  
perform action on the application

↓  
enter actual result in the  
expected actual result if  
the actual & expected result  
are same.

update the status as passed -

If the actual & expected result are different  
than update the result as failed.

↓  
Repeat the above step for all possible  
scenarios.

\*  
finally click on end run icon

### Resume points:

- Extensively work on test management tool ALM
- Expertise in writing test cases & execution of test cases in ALM
- Proficient in logging & tracking the defects using ALM
- Prepared requirement traceability matrix using ALM.

# White Box Testing

papergrid

Date: / /

## Definition:

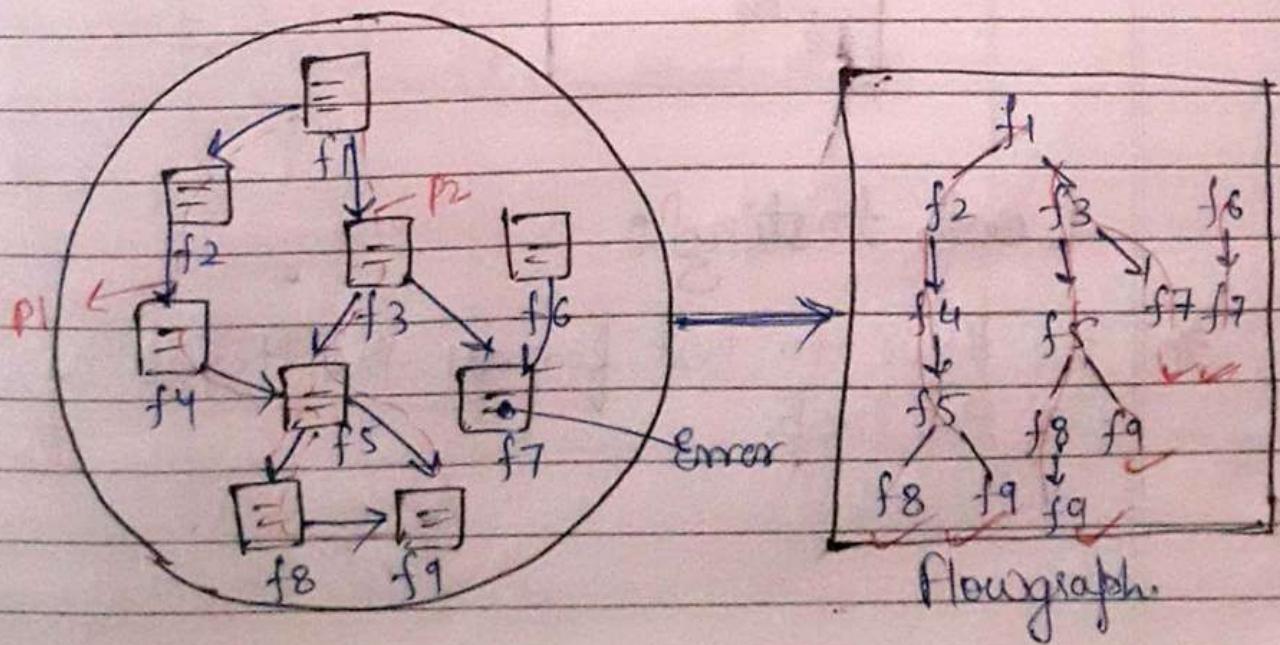
Testing each & every line of the program is called white box testing.

It is done by developer before giving the build to the testing team in order to reduce the bug count. It can be done manually or can be automated.

## Types of White box testing:

- 1) Path testing
- 2) condition testing
- 3) Loop testing.

## Path testing:



- \* Write the flow graph & test all the independent path.
- \* Whenever there is a defect fixes we should test all the dependent paths.

### Condition testing

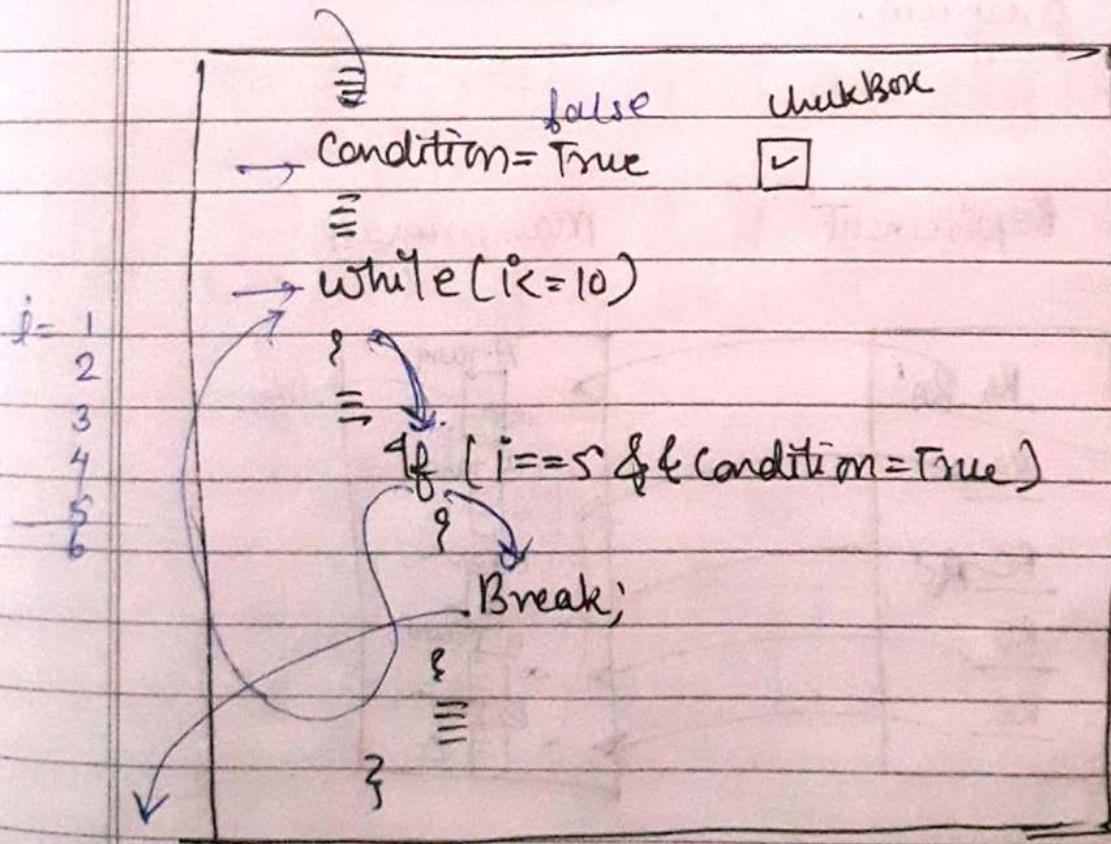
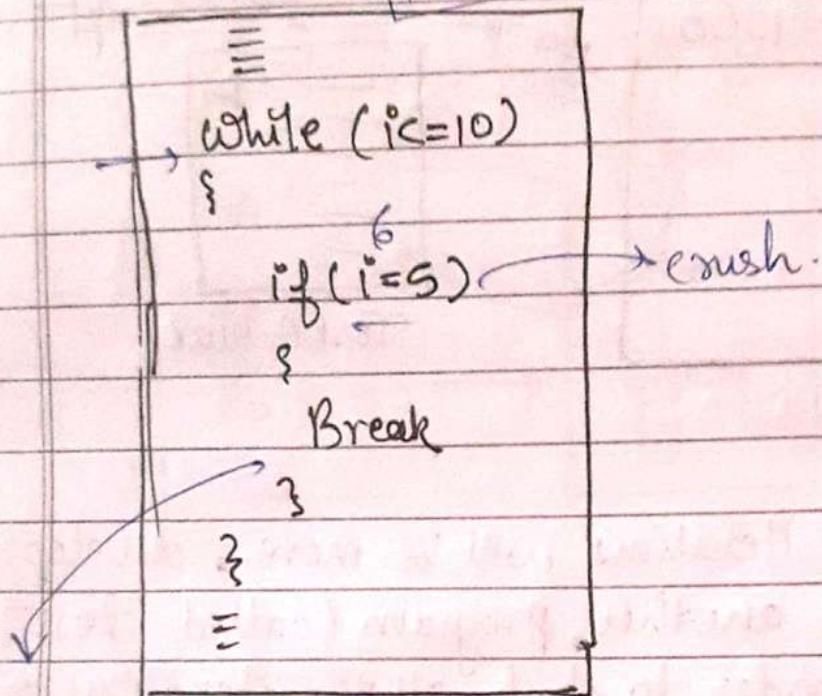
- \* When in we should test for both true & false values.

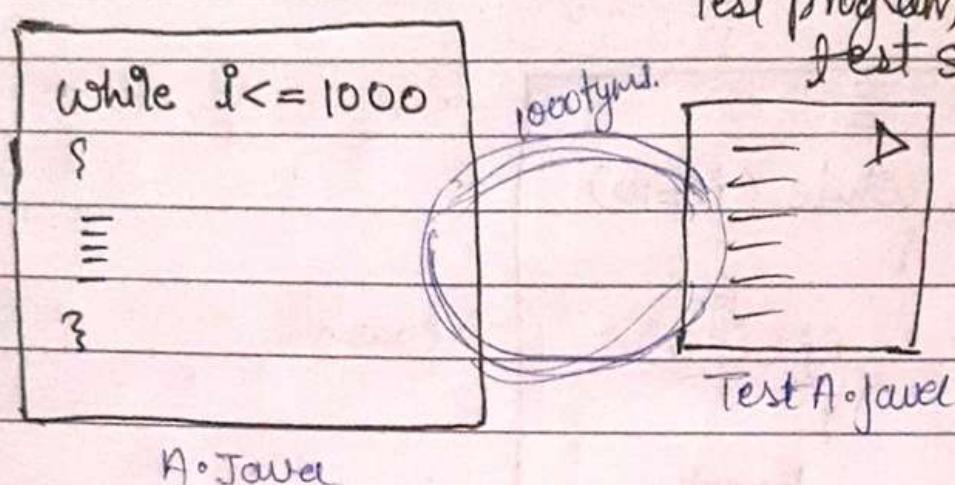
if (condition)
{
= B1 ✓
?
else
{
= B2
?
}

### Loop testing:

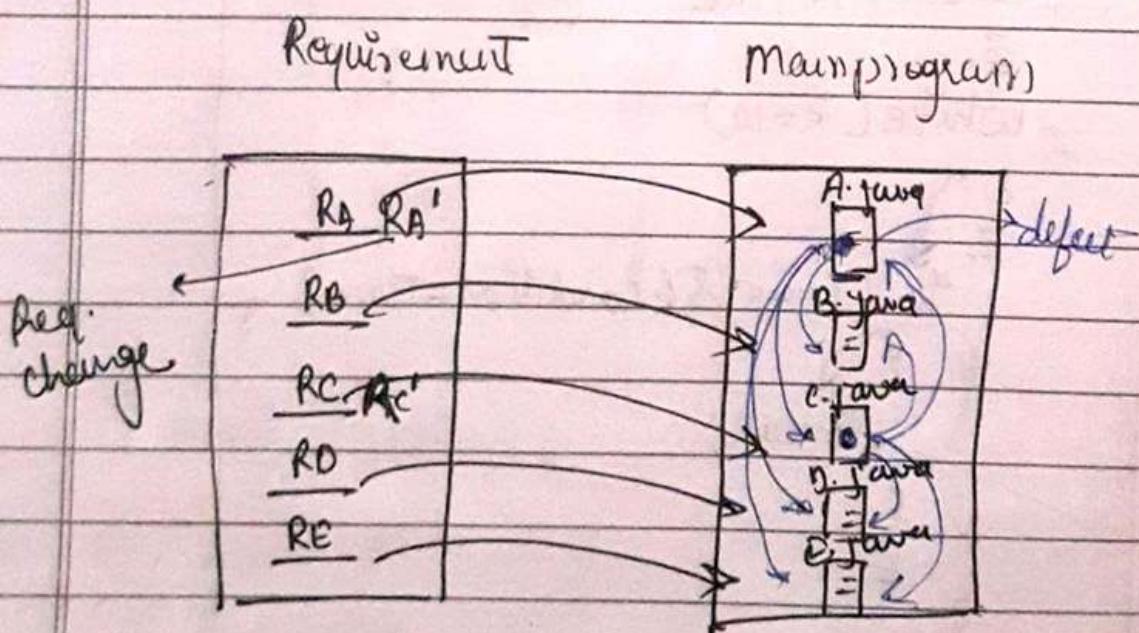
- \* We have to test for all the iterations of the loop.

11 2 3 4 5 6 7 8 9 10



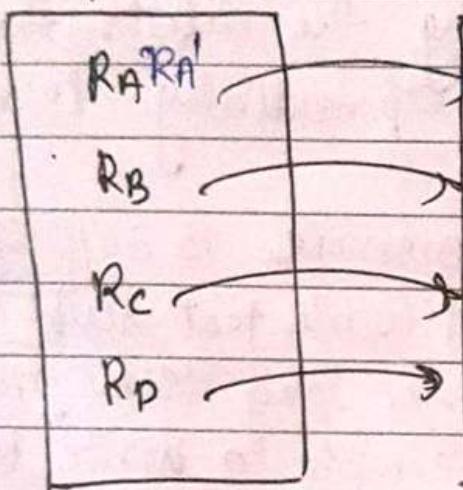


When no. of Iterations will be more, developer will write another program (called test script) in order to check all the iterations of the program.

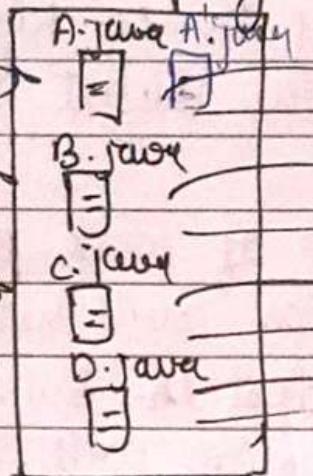


Problem: Developer will waste more time on checking the Unrelated areas

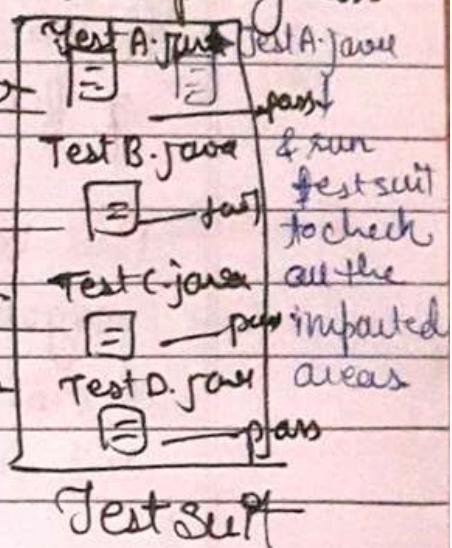
## Requirement



## Main program

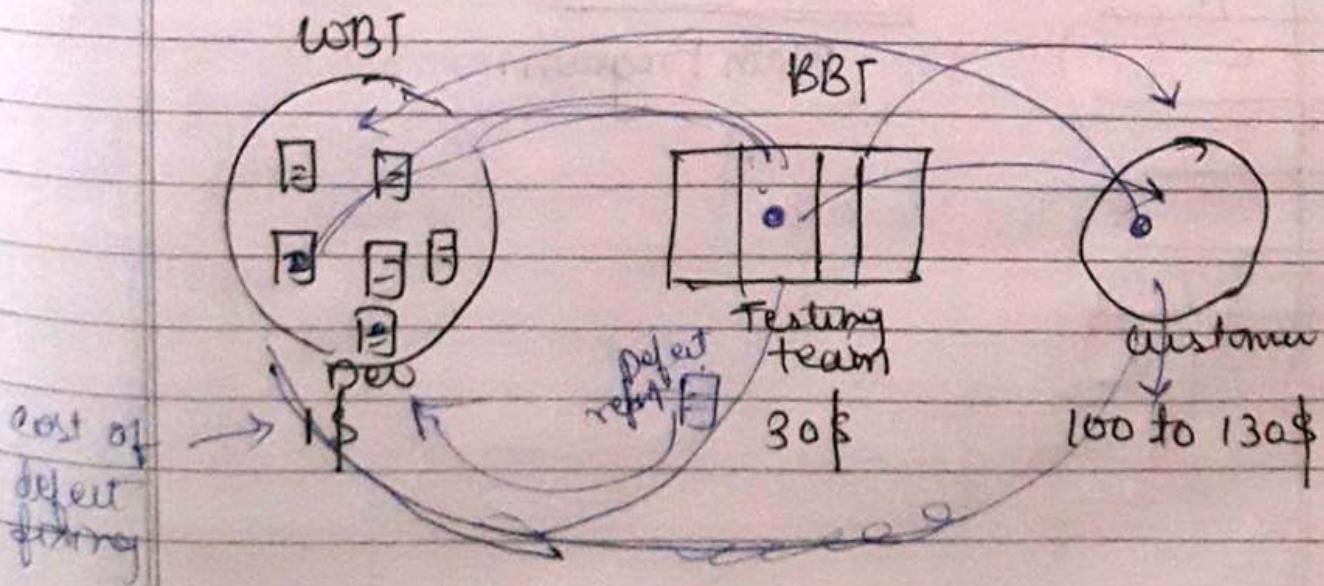


## Test program



now time will be spend on doing changes  
not on

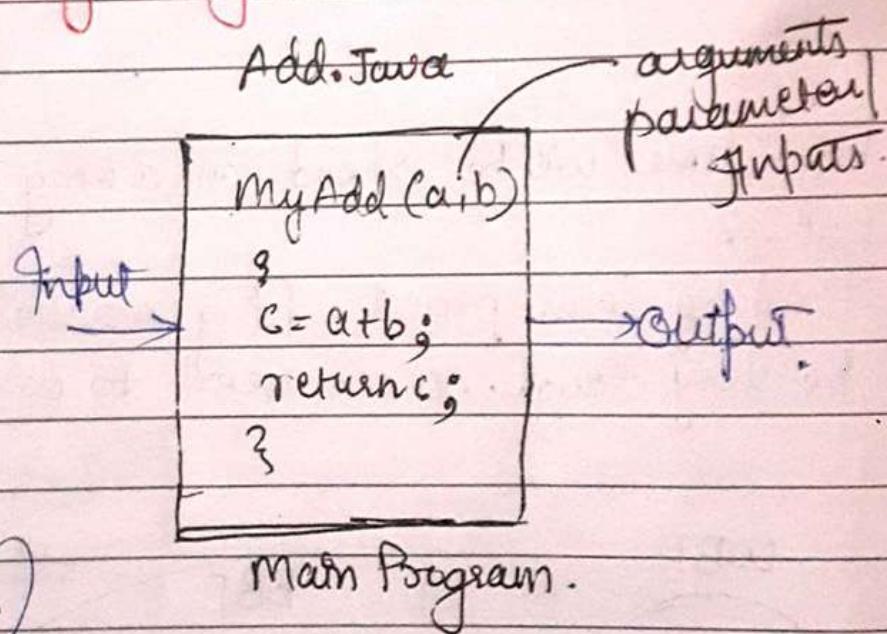
For long term project, doing manual WBT will  
be very tough. So we need to write test script.



If we delay in catching the defects the cost of fixing the defect exponentially increases.

But because of work pressure 70-80% of organisation will not write test script & so the fest is. but for long term projects 20% organisation will complete to write test script.

### Manually doing IGBT :-



Input		O/P
a	b	c
10	20	30
10	-5	5
5.2	3.5	8.7
10	A	Error

Test Scenarios

## WBT using automation:

Input	Output
a	c
10	30
10	-5
52	87
10	A

Test Scenarios.

Errors.

i/p

Add.java

```
myAdd(a, b)
{
    c = a + b;
    return c;
}
```

TestAdd.java

```
x = call Add.myAdd(10, 20);
If(x == 30)
{
    print 'pass'
}
else
{
    print 'fail'
}
```

Test2Add.java

```
x = call Add.myAdd(10, -5);
If(x == 5)
{
    print 'pass'
}
else
{
    print 'fail'
}
```

Test3Add.java

```
x = call Add.myAdd(10, A)
```

If(x == Error)
 print pass

else

print fail.

compare  
(y, 5)

(matching) pass

compare  
(x, 30)

(not matching) fail

size of test prog  
is greater than main  
prog

size of test prog  
is greater than main  
prog

Test4Add.java

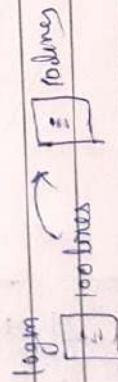
```
x = call Add.myAdd(52, 35);
If(x == 87)
{
    print pass
}
else
{
    print fail
}
```

Reducing the No. of statements without changing the output is called ~~good~~ code optimisation.

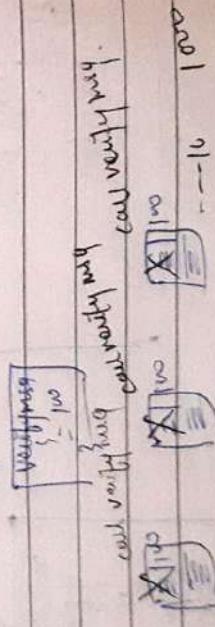
white box testing from memory point of view:

- Q. What are the typical mistakes done by the developers ~~so~~ with memory of the memory size?

- ④ Because of logical mistake.



- ⑤ Because of not using pre-built functions:  
→ my sent file is available but i am not using built-in function  
⑥ Because of not using functions for repeated steps.

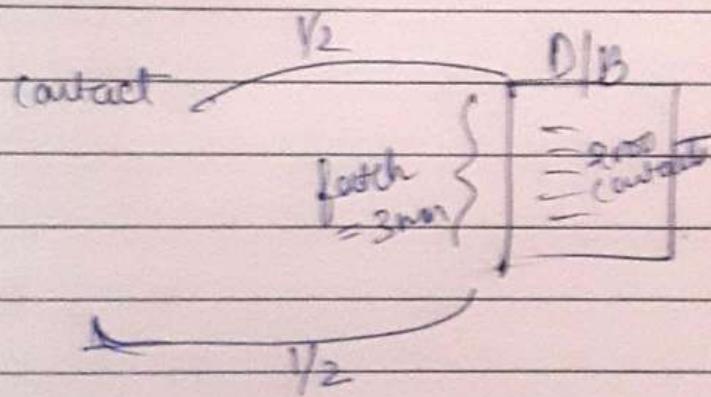


- ④ Because of unused variables, functions & objects.  
~~⑤ performance~~

White Box testing from performance point of view:

- Q. What are mistakes done by the developer which is reducing the response time?

- ① Logical mistake.



Total response = 4 min

↓ optimise

fetch only 20 contacts & display 20  
 next fetch next 20 contacts & display next 20  
~~set user cold~~ (It will take same  
 time but customer will get feel that  
 it's ~~taking~~ fast)

e.g. In amazon (keep adding new products 1 by 1)

- ②. Because of using nested condition instead of switch statement.