# Manual testing

## Contents:

# Functionality Testing
# Field Level Testing /Component Testing

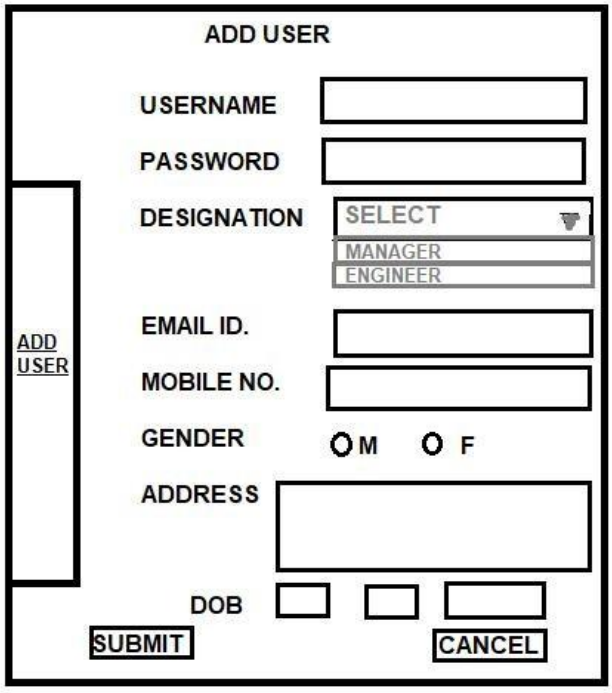
Testing each and every component thoroughly against the requirement specificationis called functionality testing

## SRS

1. Add User Link

When user clicked add user page should be displayedAnd the following should be displayed

1.1 Username textfield: Should accept 4 to 20 alphabets
1.2 Password Textfield: Should accept only 8 to 20 characters, and special charactersare allowed
1.3 Designation drop down list: is mandatory
1.4 Email ID: Should accept only valid email ID
1.5 Mobile No : Should accept only 10 digit integer
1.6 Gender radio button: is mandatory
1.7 Address text area field: Should accept only 10 to 200 characters
1.8 Date of birth : Should accept valid date of birth
1.9 Submit button: Should create a user if data is valid
1.10 Cancel Button: Should take user to home page



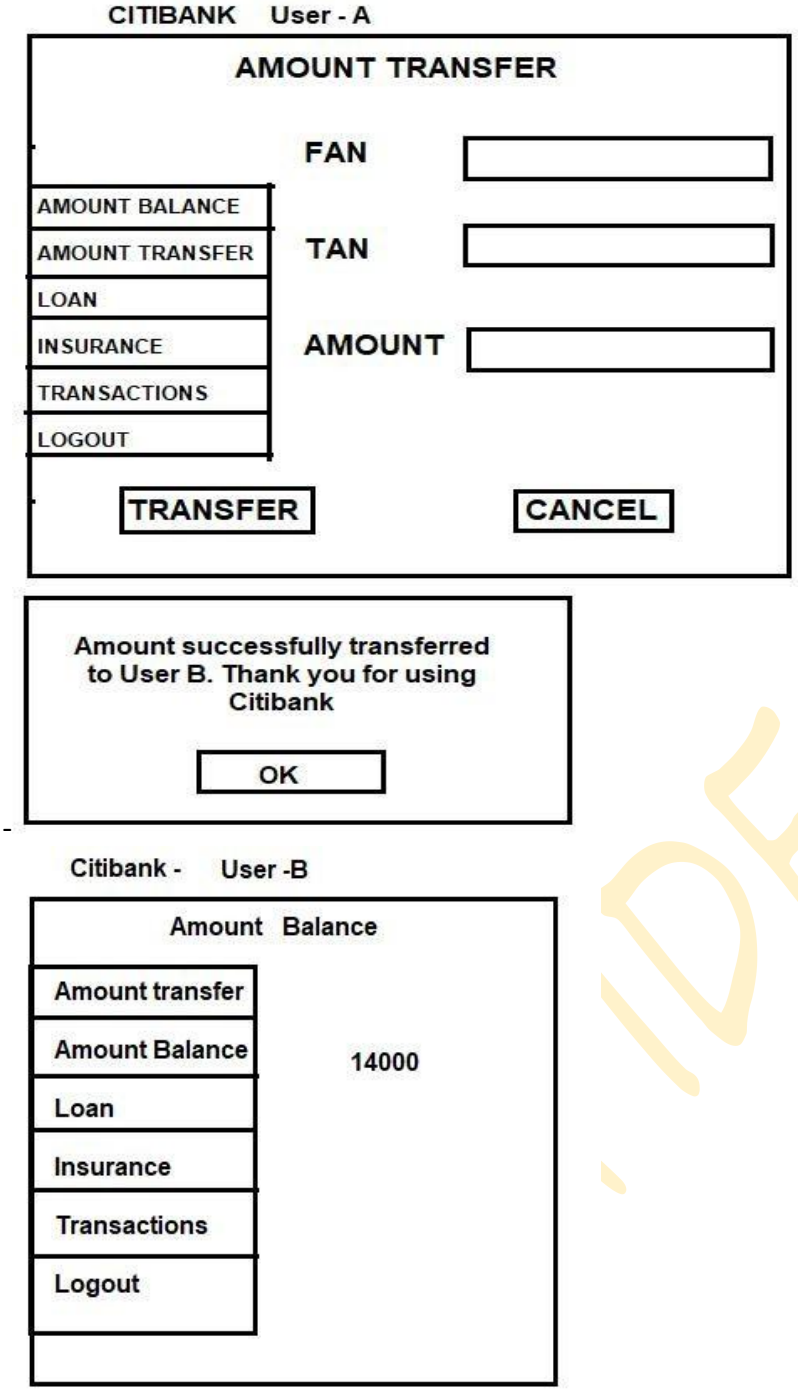

| Field Name | Input | Expected Result |
|---|---|---|
| Username | | |
| Password | | |
| Email ID | abc@gmail.com<br>abc@.com<br>abc.com<br>abcgmail.com<br>abc@gmailcom<br>abc@gmail.con<br>abc@gmail<br>abc@jmail.com | Accept<br>Error<br>Error<br>Error<br>Error<br>Error<br>Error<br>Error |

| | | |
|---|---|---|
| Mobile number | 10 digit<br>9 digit Characters<br>Special Character<br>Combination of number, character and special character | Accept<br>Error<br>Error<br>Error |
| Gender | Select male or female<br>Select male and female<br>Don't select male or female | Accept<br>Error<br>Error |

Component means it can be link, radio button, text field, text area field, drop downlist, button.
Note: Always start testing with valid data

## SRS

1. Welcome Page
1.1 Login
1.1.1 Username text field should accept 6 -32 characters
1.1.2 Password Text field should accept 4 -10 characters
1.2 Forgot password
1.2.1 .....
1.2.2 .....
2. Loan
2.1 Home Loan
2.1.1 Fixed Interest
2.1.1.1 .....
2.1.1.2 .....
2.2 Personal Loan
2.2.1 .....
2.2.2 .....
3. Insurance
3.1 .....
3.2 .....
-
-
-
28.
29. Amount Balance
30. Amount Transfer
    30.1 FAN Text field30.1.1..... 30.1.2.....
    30.2 TAN Text Field 30.2.1..... 30.2.2.....
    30.3 Amount Text Field
    30.3.1 Should accept only +ve integers
    30.3.2 Should accept more than the amount balance
-
-
31. Transactions




## Why requirements should be numbered?

1) It is very easy to understand the requirement
2) There will be clarity in the requirement
3) Requirement becomes traceable
4) Requirement becomes measurable
5) It becomes easy to communicate between developers, testing team andcustomers

## Ways of testing

1) Over Testing
2) Under testing
3) Optimized testing

### Over testing:

Testing the application with the same scenarios in different waysOR

Testing the application with those scenarios which does not make sense is calledover testing. If we do this testing, it is waste of time.

### Under Testing

Testing the application with insufficient set of scenarios is called as under testing.If we do this testing, we will miss a lot of bugs

### Optimized testing

Testing the application only with those scenarios which make sense is called asoptimized testing. If we do this testing, we are going to ind maximum bugs in optimal time

## Positive Testing

Entering valid or expected data and testing the application is called positivetesting

## Negative Testing

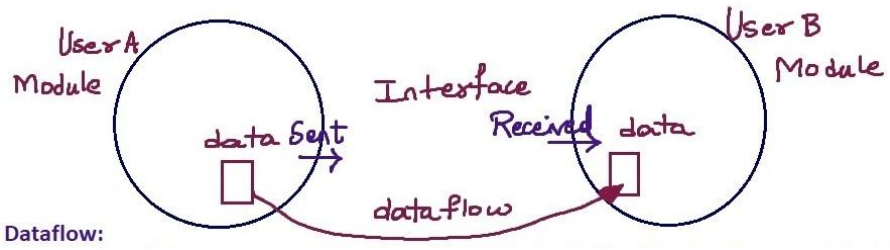
Entering invalid or unexpected data and testing the application is called negativetesting

## Rules of Testing

1) We should always do positive testing and then only negative testing
2) If application is working for positive/valid data, then only test orinvalid/negative data
3) If application is not working for one invalid input,then don't stop testing, testfor more invalid values
4) As a test Engineer we should never assume the requirements

# INTEGRATION TESTING

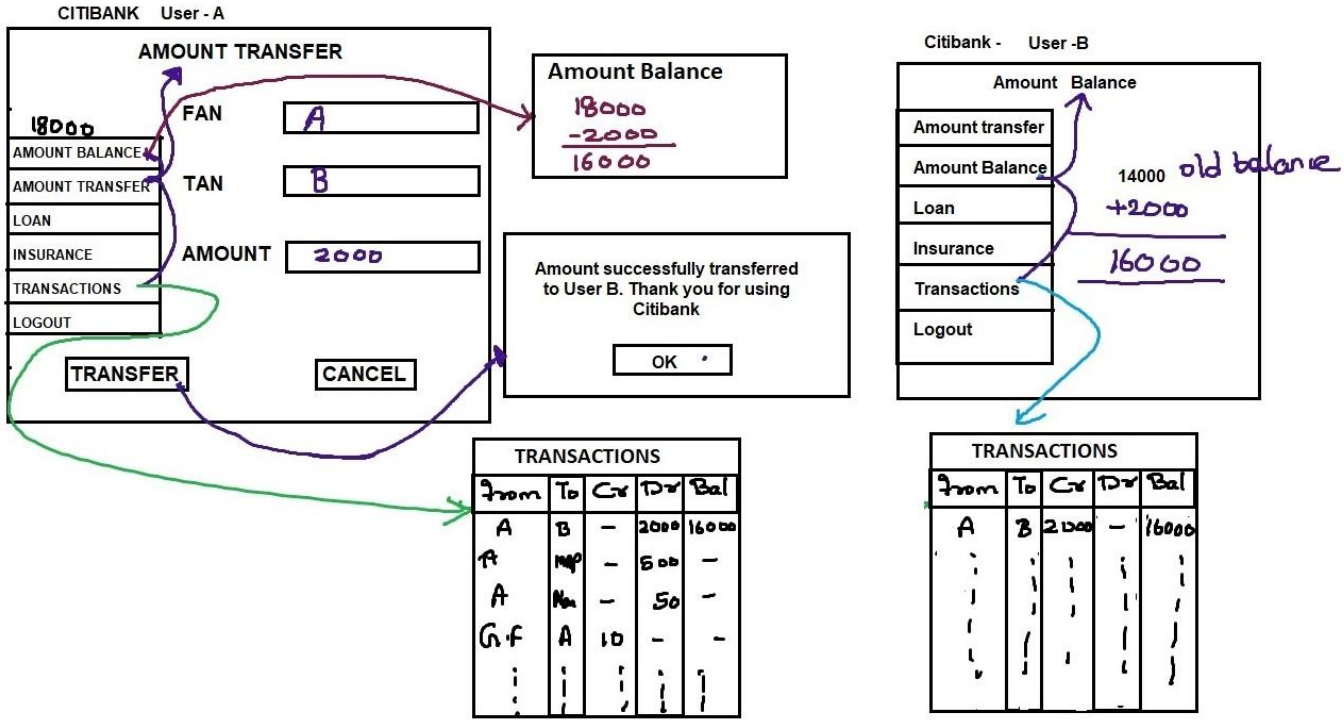
Testing the dataflow or interface between two modules is called as integration testing




## Data Flow:

Testing between two modules A and B, to check whether the data from module A is sent to module B and also testing whether the data from module A is received by module B

## Example: 1




**Dataflow-1**

1) Login as User A
2) Click on Amount Transfer
3) Enter FAN, TAN, and amount
4) Click on Transfer
5) Confirmation page is displayed, click on OK
6) Logout as User A and login as User B
7) Click on Amount Balance

**Expected Result: New balance should be displayed**

**Dataflow-2**

1) Login as User A
2) Click on Amount Transfer
3) Enter FAN, TAN, and amount
4) Click on Transfer
5) Confirmation page is displayed, click on OK
6) Click on Amount Balance

**Expected Result: New balance should be displayed**

## How to do integration testing?

1. First understand the requirement
    a) You should understand each and every module in depth
    b) You should also understand how each modules are related to each other
2. Identify all the possible scenarios
3. Prioritize the identified scenarios
4. Document the scenarios according to the priority
5. Execute the scenarios and find the defect and communicate to the developers

## Positive integration testing

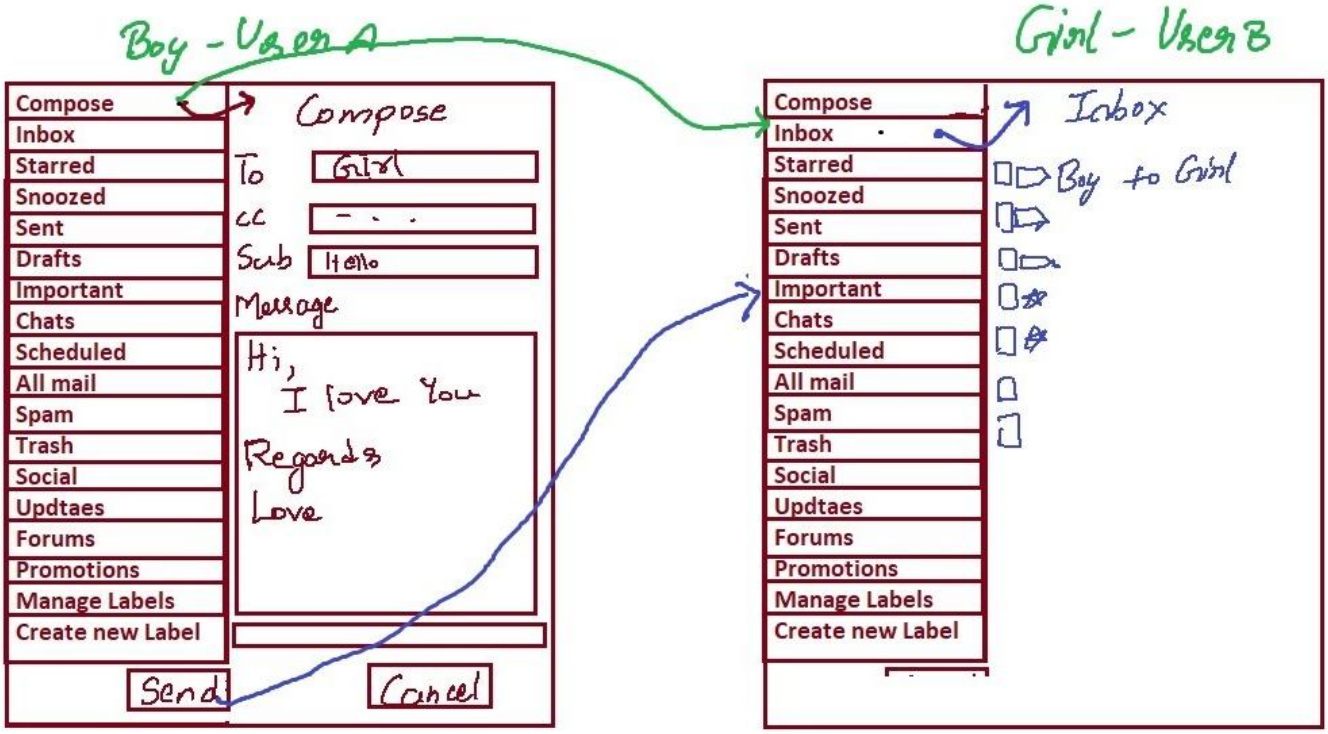
Testing the dataflow by giving valid inputs is called as positive integration testing

## Negative integration testing

Testing the dataflow by giving invalid inputs is called as negative integration testing

**Note:** If the amount transfer from A to B is tested, then no need to test the same from B to A again because the same code is tested. The code code will be saved in the server and it uses the same code again.

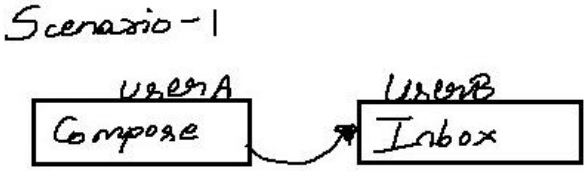
## Example : 2



**Scenario-1**

1) Login as User A
2) Click on Compose button
3) Enter all the details (To -User B, Sub, Message....)
4) Click on Send
5) Logout as User A and login as User B
6) Click on Inbox button

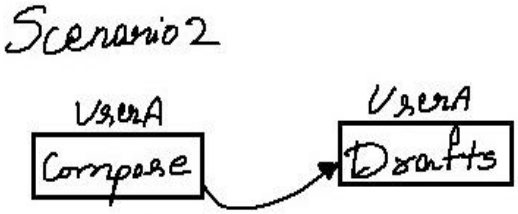
**Expected Result: The sent mail should be present**




**Scenario-2**

1) Login as User A
2) Click on Compose button
3) Enter all the details (To -User B, Sub, Message....)
4) Click on Cancel
5) Click on Drafts button

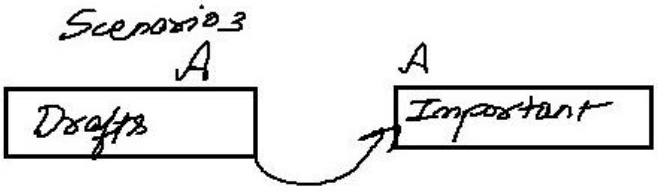
**Expected Result: The cancelled mail should be present**




**Scenario-3**

1) Login as User A
2) Click on Drafts
3) Select a mail from the list
4) Click on the kebab menu
5) Select "Mark as Important"
6) Navigate to Important

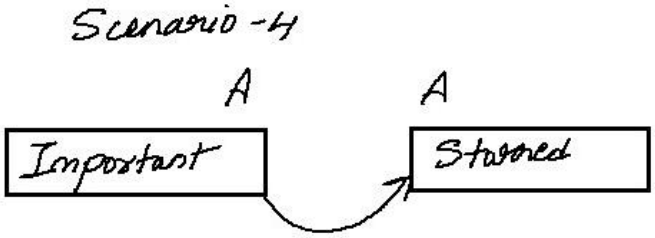
**Expected Result: The mail marked as important should be displayed**




**Scenario-4**

1) Login as User A
2) Click on Important
3) CLick on the star icon beside the mail in mail list
4) Navigate to Starred

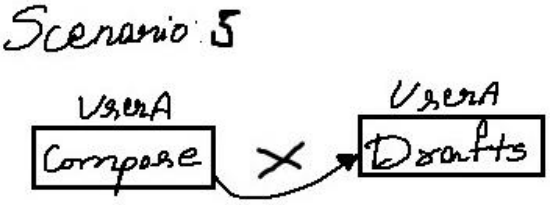
**Expected Result: The mail marked as starred should be displayed**




**Scenario- 5**

1) Login as User A
2) Click on Compose button
3) Enter all the details (To -User B, Sub, Message....)
4) Click on send
5) Click on Drafts button

**Expected Result: The sent mail should not be displayed in drafts**




## Types of Integration

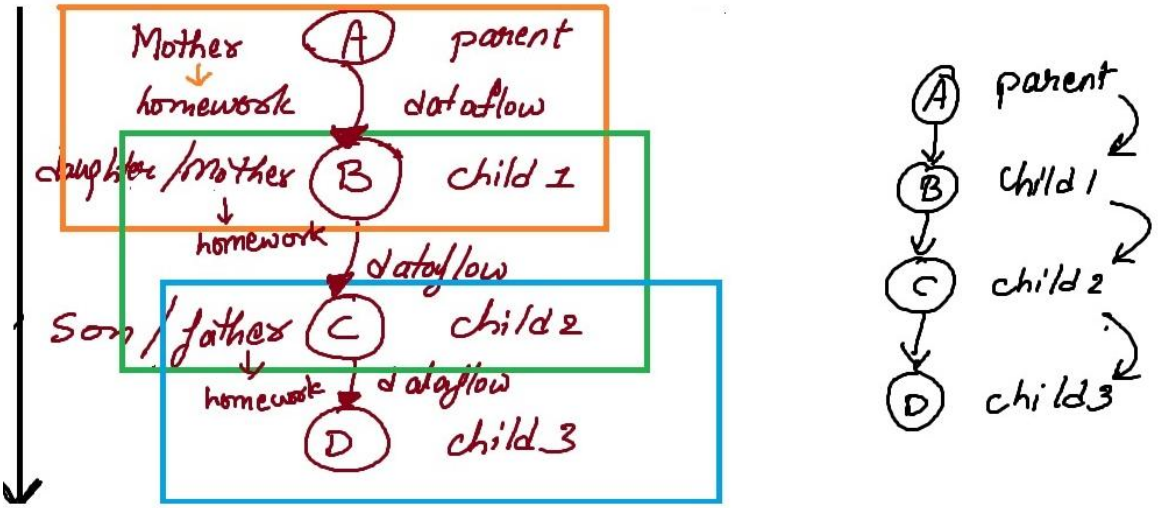
1. Incremental Integration Testing
    a) Top Down Incremental Integration Testing
    b) Bottom Up Incremental Integration Testing
2. Non- Incremental Integration Testing

# Incremental Integration Testing

Incrementally add the modules and test the dataflow between the modules

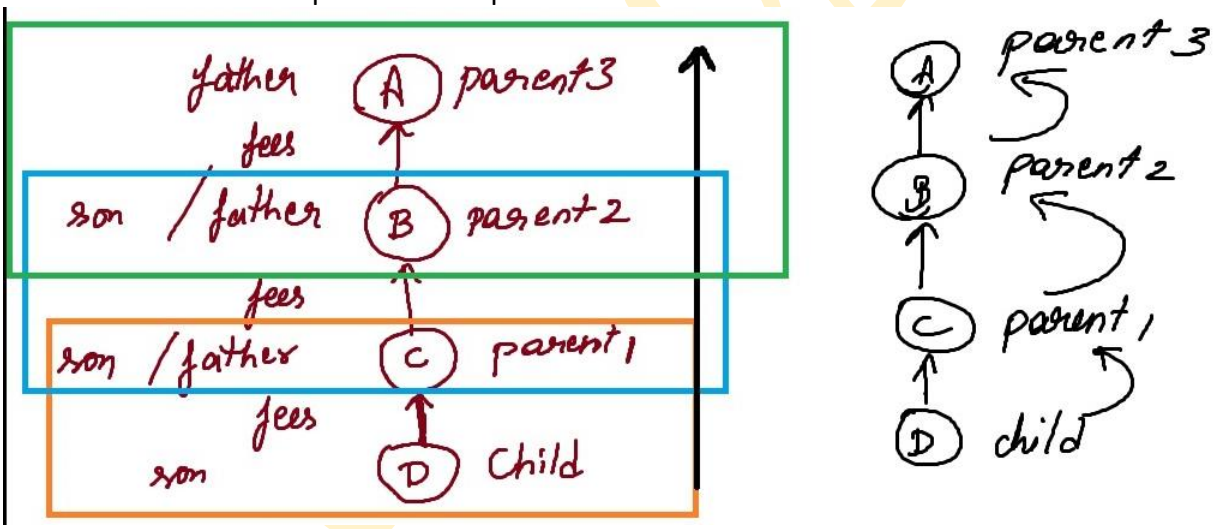
**a) Top Down Incremental Integration Testing**

Incrementally add the modules and test the dataflow between the modules. Make sure that the modules added are the child of the previous modules




**b) Bottom Up Incremental Integration Testing**

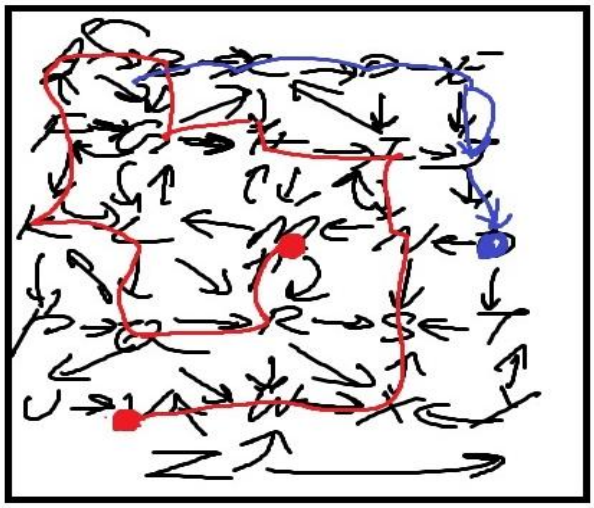
Incrementally add the modules and test the dataflow between the modules. Make sure that the modules added are the parent of the previous modules




**Note:** If there are n number of modules it is difficult to identify whether who is parent and child. So we go for NIIT.

# Non- Incremental Integration Testing

Combine all the modules at a short and test the dataflow between the modules

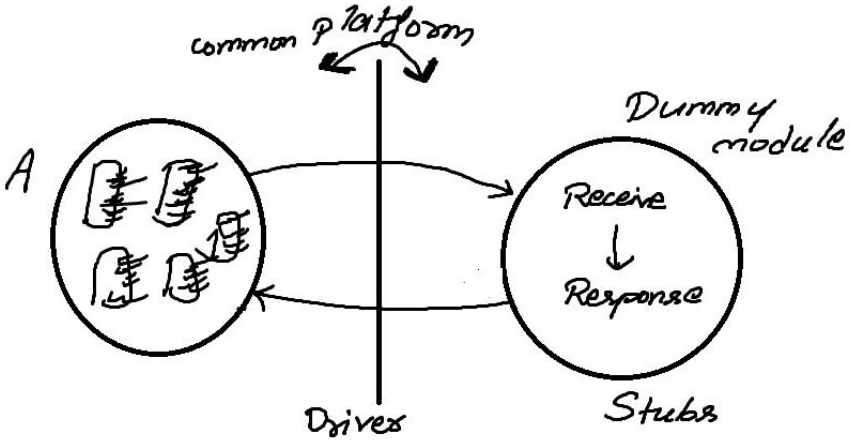


**Drawbacks:**

1. We might miss to test certain scenarios
2. Finding the root cause might be difficult

**Note:** We do non-incremental integration testing when we don't have connection between parent and child

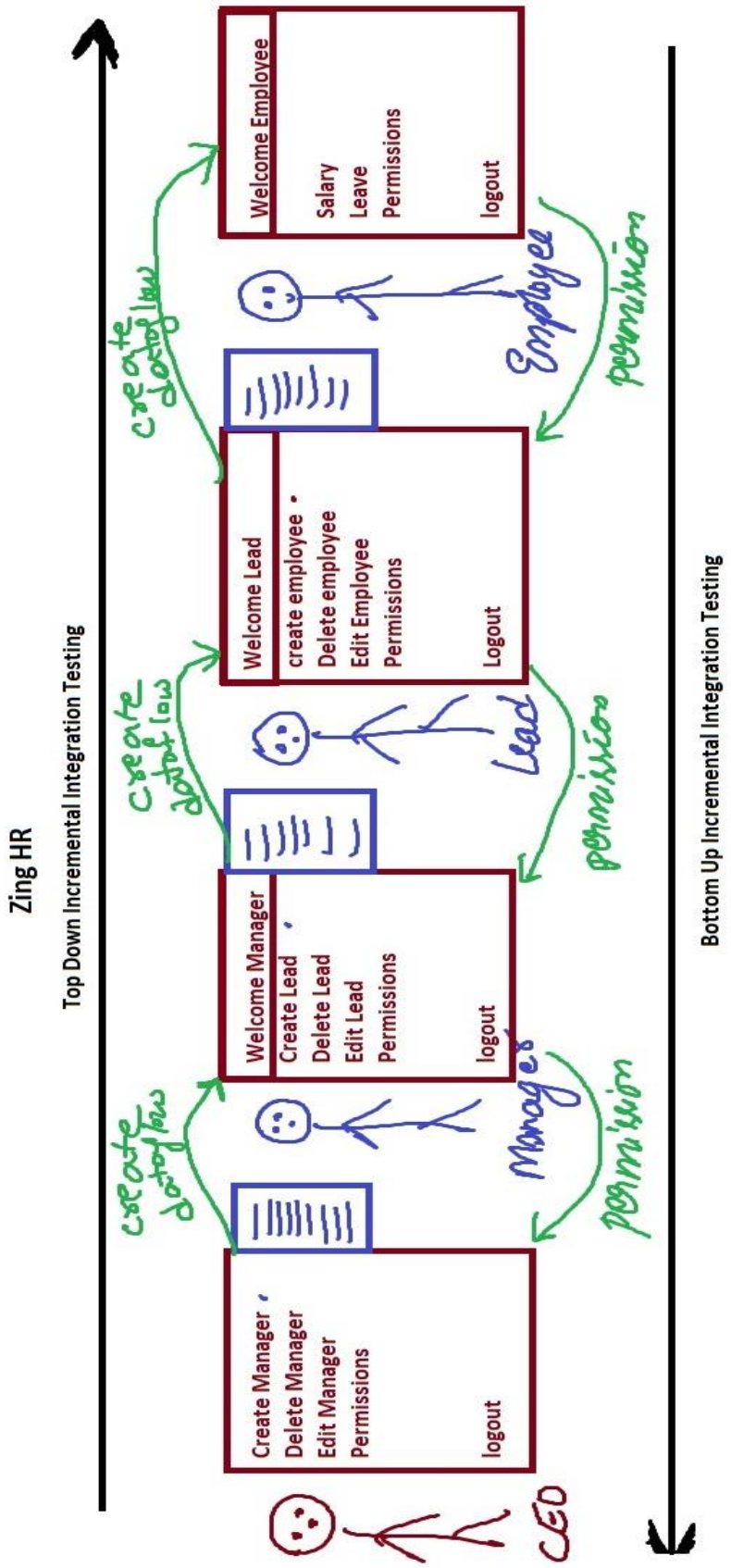
## Difference between stubs and drivers




If one module is ready and another module is not ready, then how will you do integration testing?

**Stubs:** It is a dummy module. It acts like a module which is not yet built. It can generate data and receive the data

**Drivers:** It is one which sets up the testing environment and does lot of transactions, analyse the result and sends the output. (In other words, it does the transaction between real modules and stubs)

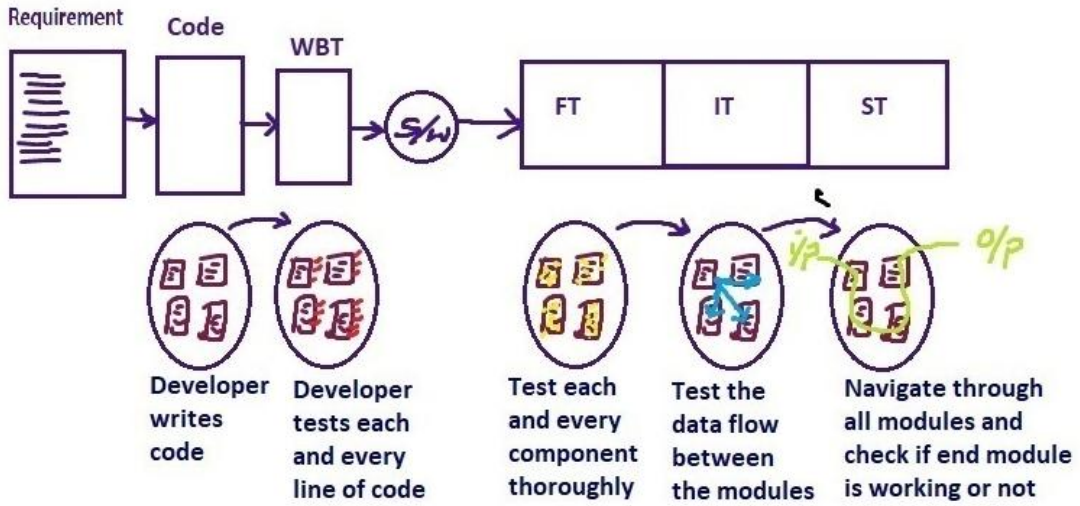

Zing HR
Top Down Incremental Integration Testing
Bottom Up Incremental Integration Testing
Welcome Employee
Salary
Leave
Permissions
logout
Welcome Lead
create employee ,
Delete employee
Edit Employee
Permissions
Logout
Welcome Manager
Create Lead ,
Delete Lead
Edit Lead
Permissions
logout
Create Manager ,
Delete Manager
Edit Manager
Permissions
logout
create dotnet labs
create dotnet labs
create dotnet labs
Employee permission
Lead permission
Manager permission
CEO
Manager
Lead
Employee

# SYSTEM TESTING

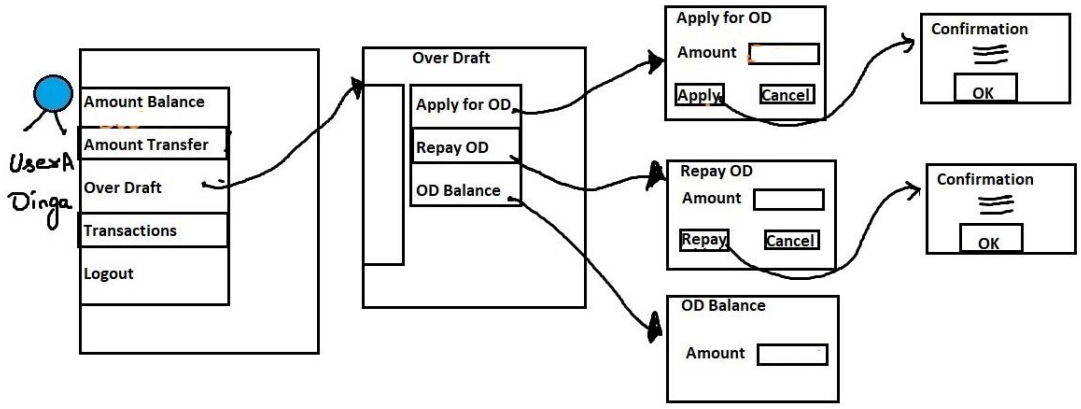
It is an End to End testing where in the testing environment is just similar to production environment

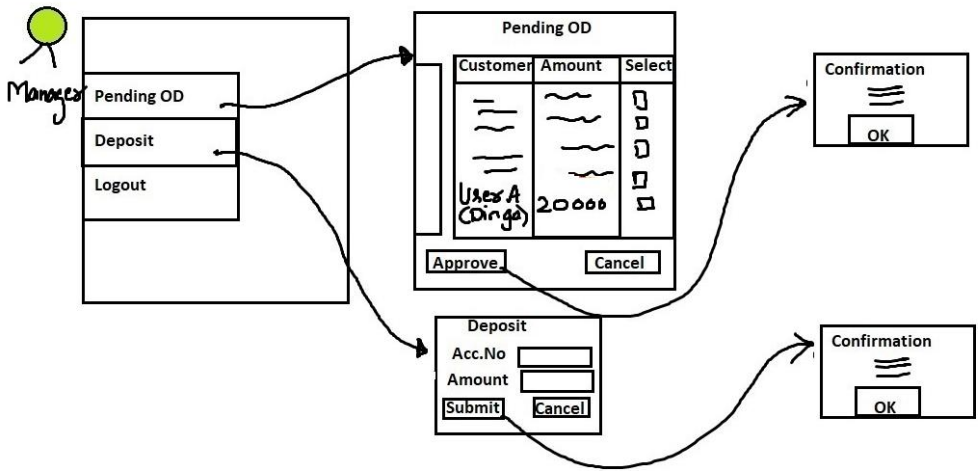
## End to End Testing

Navigating through all the features and checking whether the end features are working or not. Here we are not worried about each and every component and the dataflow




## OD flow




## Scenario

1. Login as user A
2. Click on OD button, OD page should be displayed
3. Click on Apply for OD button, Apply for OD page should be displayed
4. Enter the amount 20000 and click on Apply button

5. Confirmation page should be displayed, click on OK
6. Logout as User A and login as Manager
7. Click on Pending OD button, pending OD page will be displayed
8. Select User A and click on Approve
9. Confirmation page should be displayed, click on OK
    a) Note: As soon as the OD is approved, three things should happen
        i. Amount balance of USer A should be 20000
        ii. OD balance of User A should be 20000
        iii. And interest for the OD balance should be added on daily basis
10. Logout as Manager and Login as User A and Navigate to OD, OD page should be displayed
11. Click on OD balance, OD balance should be 20000
12. Click on Amount balance, Amount balance should be 20000
13. Change the server time to one month later
14. Click on OD balance, balance should be 20650
15. Logout as User A and login as Manager, click on deposit
16. Deposit page should be displayed, enter the User A account number , amount as 650 and click on submit, confirmation page should be displayed. Click on ok
17. Logout as Manager, login as USer A
18. Click on OD , OD page should be displayed
19. Click on Repay OD, Repay OD page should be displayed
20. Enter the amount as 20650, click on Repay, OD should be repayed
21. Navigate to OD balance, amount should be 0
22. Repeat the steps from 1 to 13.
23. OD balance should be 20400
24. Repeat steps till 21 (replacing 20650 and 650 by 20400 and 400)

# Types of Environment

## Development Environment

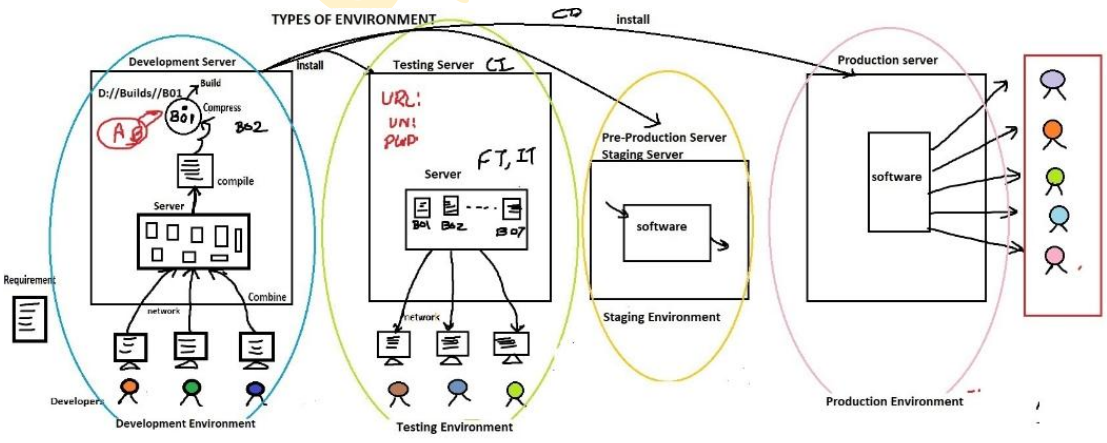
It is a setup which is used for developing the software. It contains hardware, software, server and network

## Test Environment

It is a setup which is used for testing the software. It contains  hardware, software, server and network

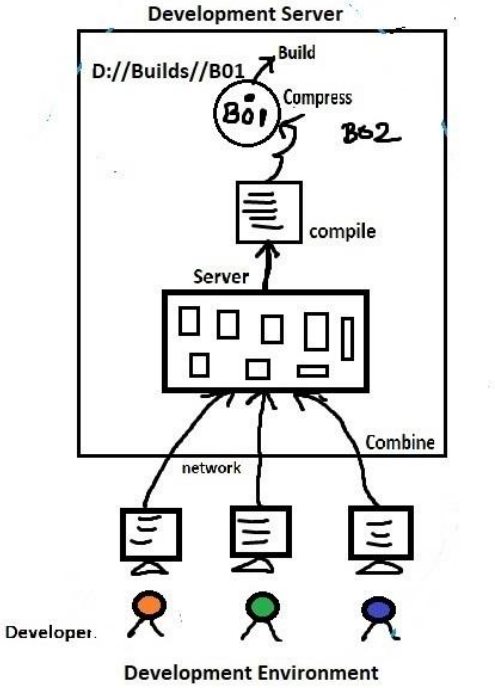
## Production Environment

It is a setup which is used to run the software for the business.  It contains  hardware, software, server and network



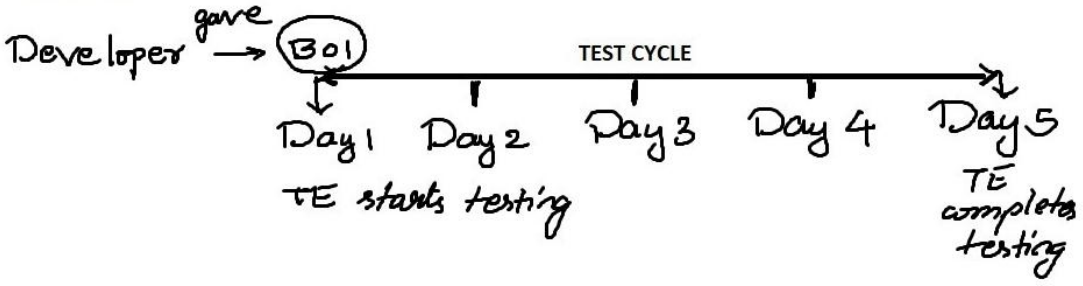

# Build:

All the programs are combined, compiled and then we get binaries and all the binaries are archived/compressed in to a file format called as build



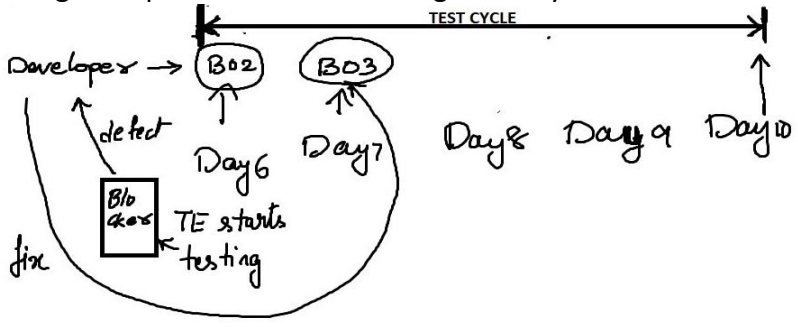

## Test cycle:

It is an effort or time or duration spent to start and finish the testing of complete build
OR
The total time taken to test one build is called as test cycle
Each test cycle might take 3, 5 or 15 days based on
a. No of Engineers
b. Complexity of software
c. Size of the product
**Note:** Every bug fix we will not get a new build. If we get, there is lot of time waste in installation and uninstallation of the build




## Respin:

Getting multiple builds with in a single test cycle is called as Respin



## Who will be involved in installation of the software?

1) Anybody from the testing team
2) Anybody from the development team
3) Release Engineer or Build Engineer

**Note:** Release Engineer manages the code, compiles, compresses the code, installs and releases to the production. This operation is called as development operation

## How the build mail comes from the development team?

Hi team
New build B01 is ready for testing
Test URL: https://qa.bank.com
UN: 123
PWD: abc
Happy testing

## Who is the owner of the bug?

One who finds the bug is the owner of the bug

## Why we find new bugs in the old module?

1) Chances are there adding new features would have introduced a bug in the old modules
2) Fixing one bug would have introduced more bugs
3) Chances are there we would have missed it in previous cycle and finding it in the current cycle

## When do we do system testing?

1) When all the basic features are ready
2) If the environment similar to production environment is ready
3) When minimum bunch of modules are ready

# Role of Release Engineer/Build Engineer

1) Maintain the source code
2) Create the build
3) Install the build in Test Environment
4) Release the product to the production

**Version control Tool:** Used to maintain and manage the code written by the developers
**Maven:** To compile and compress the code and create the build
**Jenkins:** For the installation of the build
And for **continuous integration and continuous deployment**

## What is Release?

Starting from the collecting the requirement, developers writing the code followed by WBT, and TE testing the application for many cycles, until we release the product to the production is called as 1 Release

**Note:**
1 Release might generally take 6 months to 1.5 years
If it is Agile, it might take 1 to 1.5 months

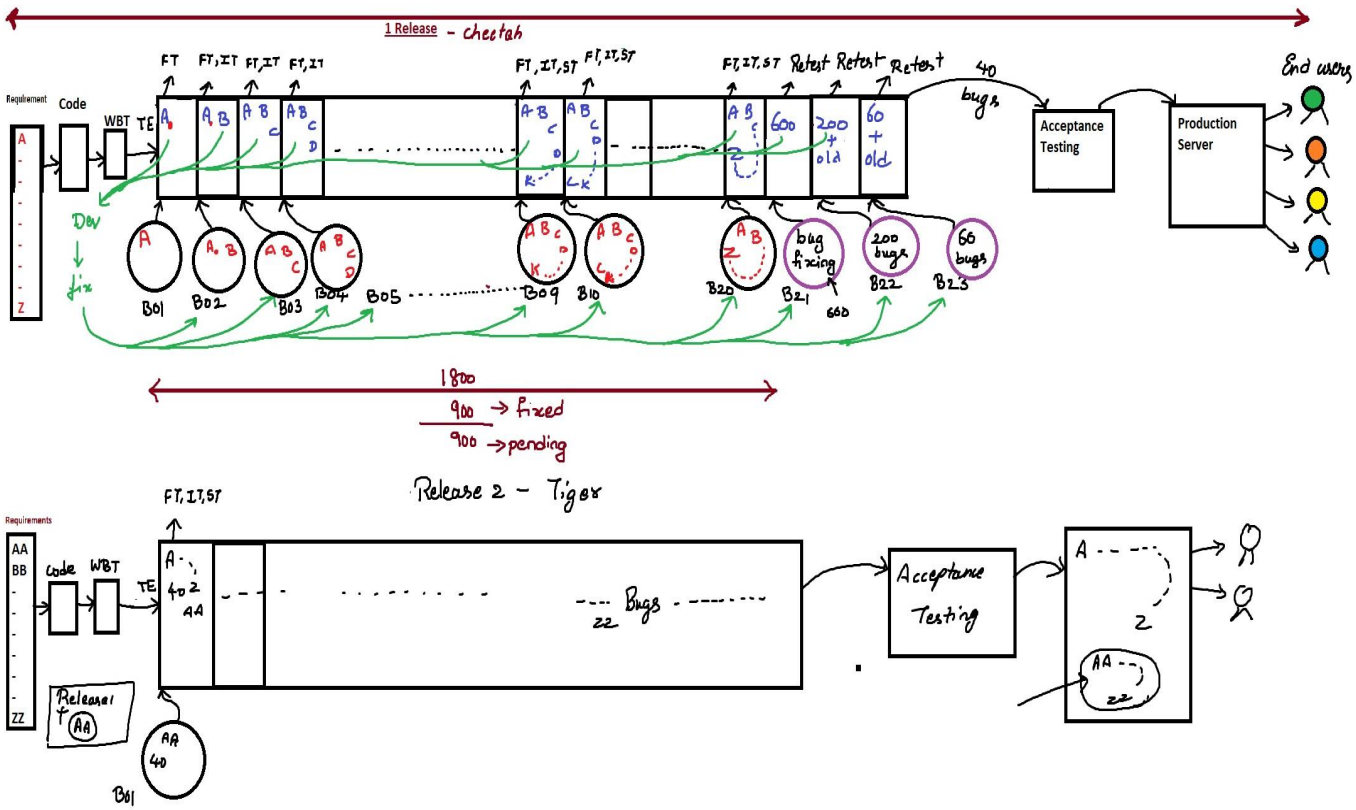
## When do you release the product to the production?

1) When there are no blockers or critical bugs
2) When all the end to end business scenarios are working fine

3) When all the features requested by the customer are ready
4) If the product is tested in the environment similar to production environment

**Note:**

1) 1 project can have only 1 release -> Small Projects
2) 1 project can have n releases -> huge projects
3) 1 release can have n number of builds and hence n number of test cycles




## Types of Application

1) Web Based Application
2) Client Server based application
3) Desktop Application/Standalone application

### Web Based Application

Any application that can be accessed by launching or opening the browser and entering the URL is called as web based application

Ex: Gmail, Facebook, Yahoo

### Client Server Application

Here 2 softwares are required, that is client software and the server software and both the softwares will interact with each other to display the application

Ex: Facebook, Whatsapp

### Desktop Application/Stand Alone application

Here we take .exe files and install the software to the system(Computer/laptop). Andonly one person can use the software at a time. This works without the internet

Ex: Notepad, Ms word