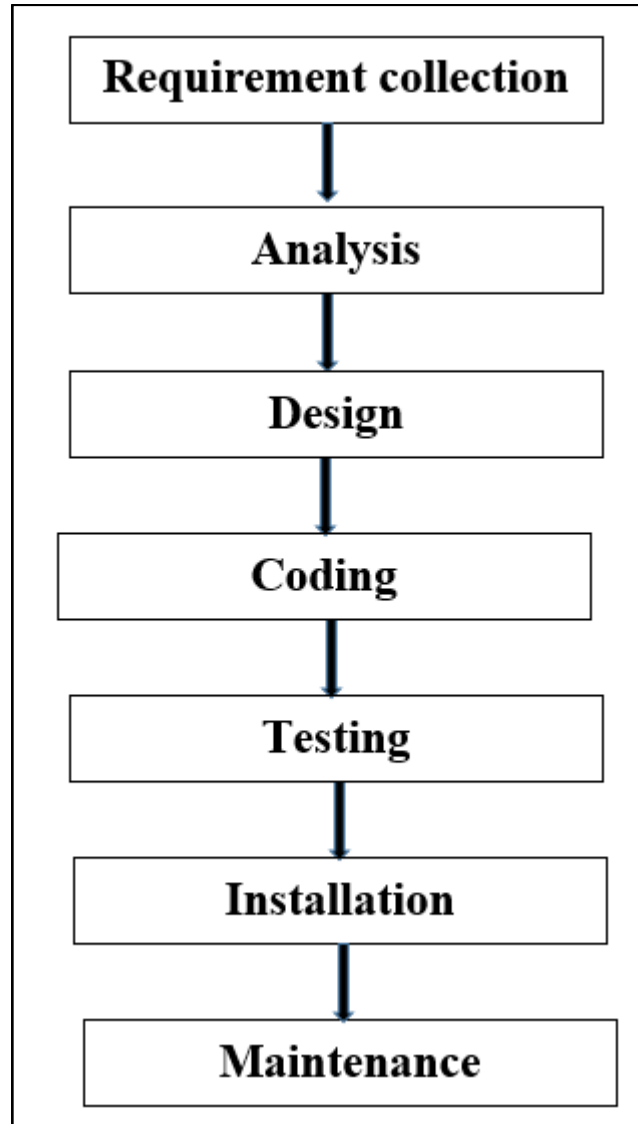


Manual Testing

SDLC: - Software Development Life Cycle

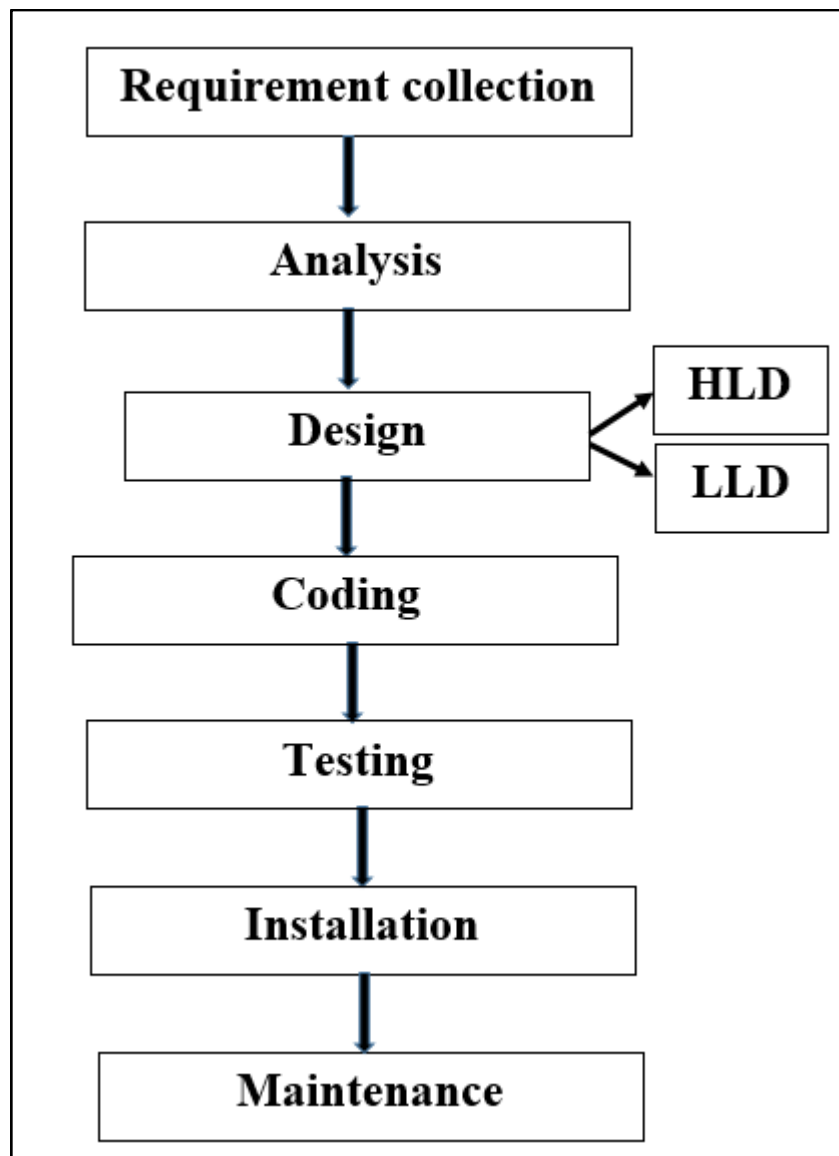
- **SDLC** - It is the procedure to develop the Software.
- Stages of Software Development Life Cycle are:



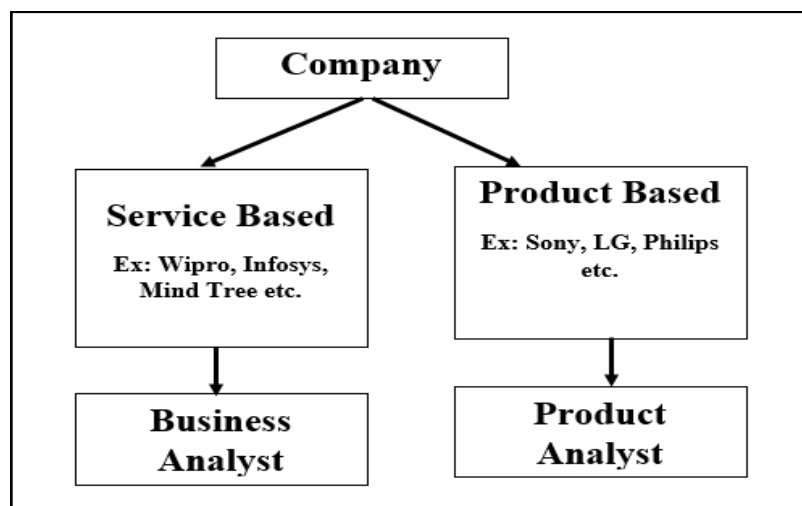
- **Different Models:**

- 1) Waterfall Model.
- 2) Spiral Model.
- 3) V-Model.
- 4) Prototype Model.
- 5) Customised/ Derived Model.
- 6) Hybrid Model.
- 7) Agile Model.

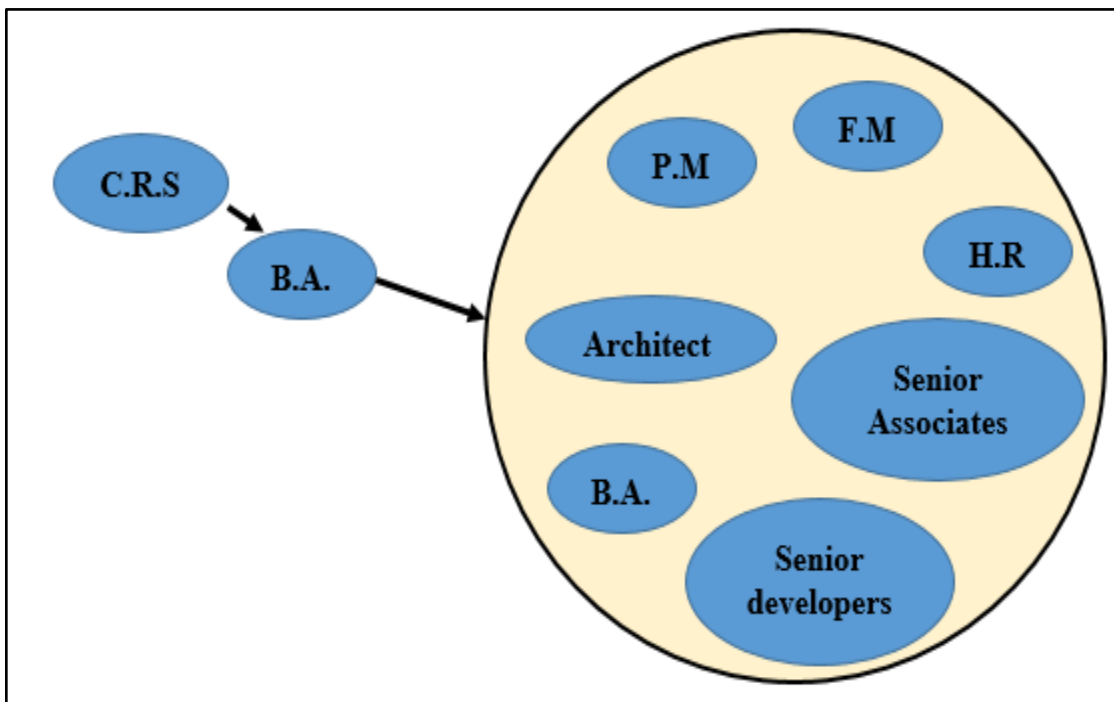
1) Waterfall Model :



- **Requirement Collection:** Basically Business Analyst (B.A) / Product Analyst (P.A) / Subject Matter Expert (S.M.E) will be involved in the requirement collection. Here they go to the customer place, collect the requirement that is CRS (Customer Requirement Specification) and will convert that CRS to SRS (Software Requirement specification).



- **Analysis (Feasibility Study):** It is that study of the project whether it is technically feasible or not, financially profitable or not and also, resources are available or not.

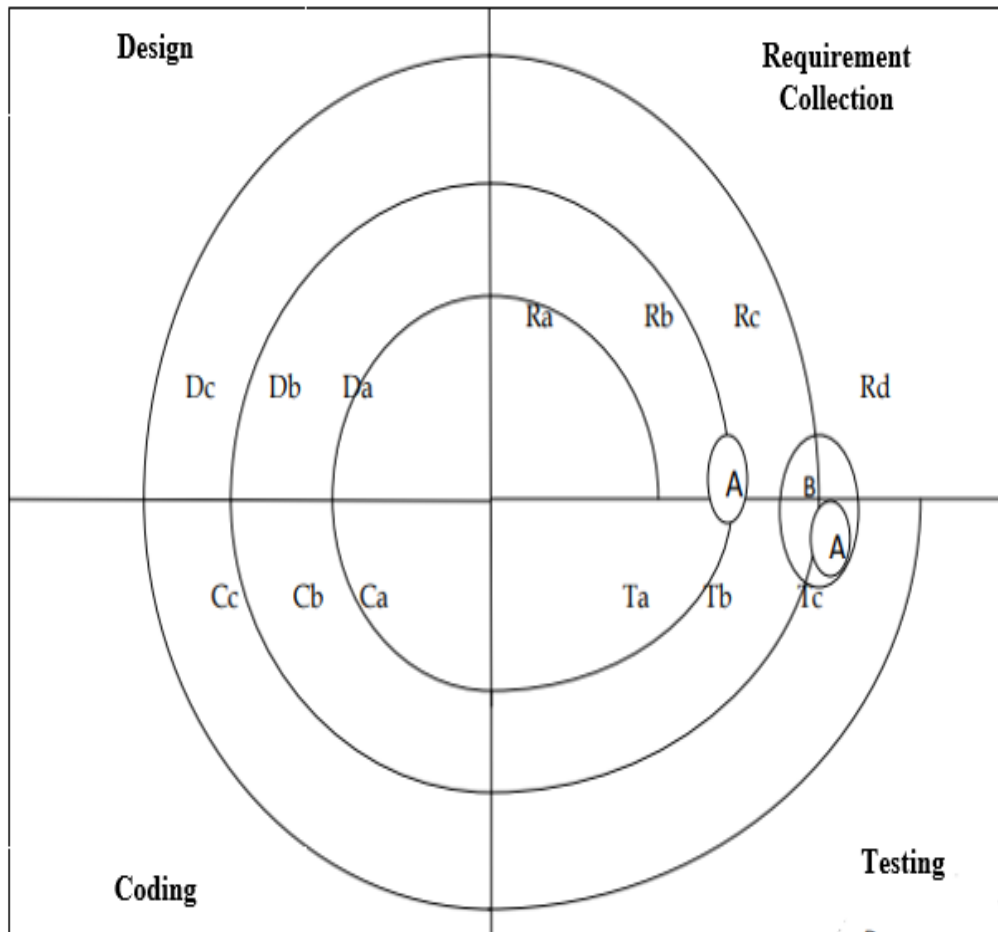


- **Design:** There are 2 stages in design,
 1. **HLD** – Gives the architecture of the software product to be developed and is done by architects and senior developers.
 2. **LLD** – Done by senior developers. It describes how each and every feature in the product should work and how every component should work. Here, only the design will be there and not the code.
- **Coding:** Developers will write the program for the modules of the software.
- **Testing:** Test Engineers will be involved in the Testing of the software.
- **Installation:** Once the software has been developed and tested, the installation of the product will be done at Client's place. Senior Developers, Release engineer's/ Field engineers, Support Engineers will be involved in the Installation Process.
- **Maintenance:** Here as the customer uses the product, he finds certain bugs and defects and sends the product back for error correction and bug fixing.
 - ★ The same Old Developers and Test Engineers who built the product will be involved here, but the team size will be small because the amount of work done will be less.
 - ★ Here minor changes like adding, deleting or modifying any small feature in the software product will be performed.
- **Advantages of Waterfall Model :**
 1. It is simple to adopt.
 2. Initial investment is less.
 3. We can expect a stable product at the end.
- **Disadvantages of Waterfall Model :**
 1. Developers will be involved in testing.
 2. It is not a Flexible model.
 3. It leads to a lot of rework if there is a bug in the design, and it flows till the end and causes a lot of re-work, because designs are not tested.
 4. Due to a lot of re-work, Total cost of the project increases.
 5. Turn around time taken to deliver the product to the customer will be more.

- **Applications of Waterfall Model :**

1. When we go for short term projects.
2. When we go for simple applications.
3. Whenever we are sure that the requirements will not change.

2) Spiral Model :



- **Advantages of Spiral Model :**

1. After we build one feature / module of the product, then only we can go on to build the next module of the product.
2. Turn around time taken to deliver the product to the customer is less, because we are not developing a full product at a time.
3. Here requirement changes are allowed.

- **Disadvantages of Spiral Model :**

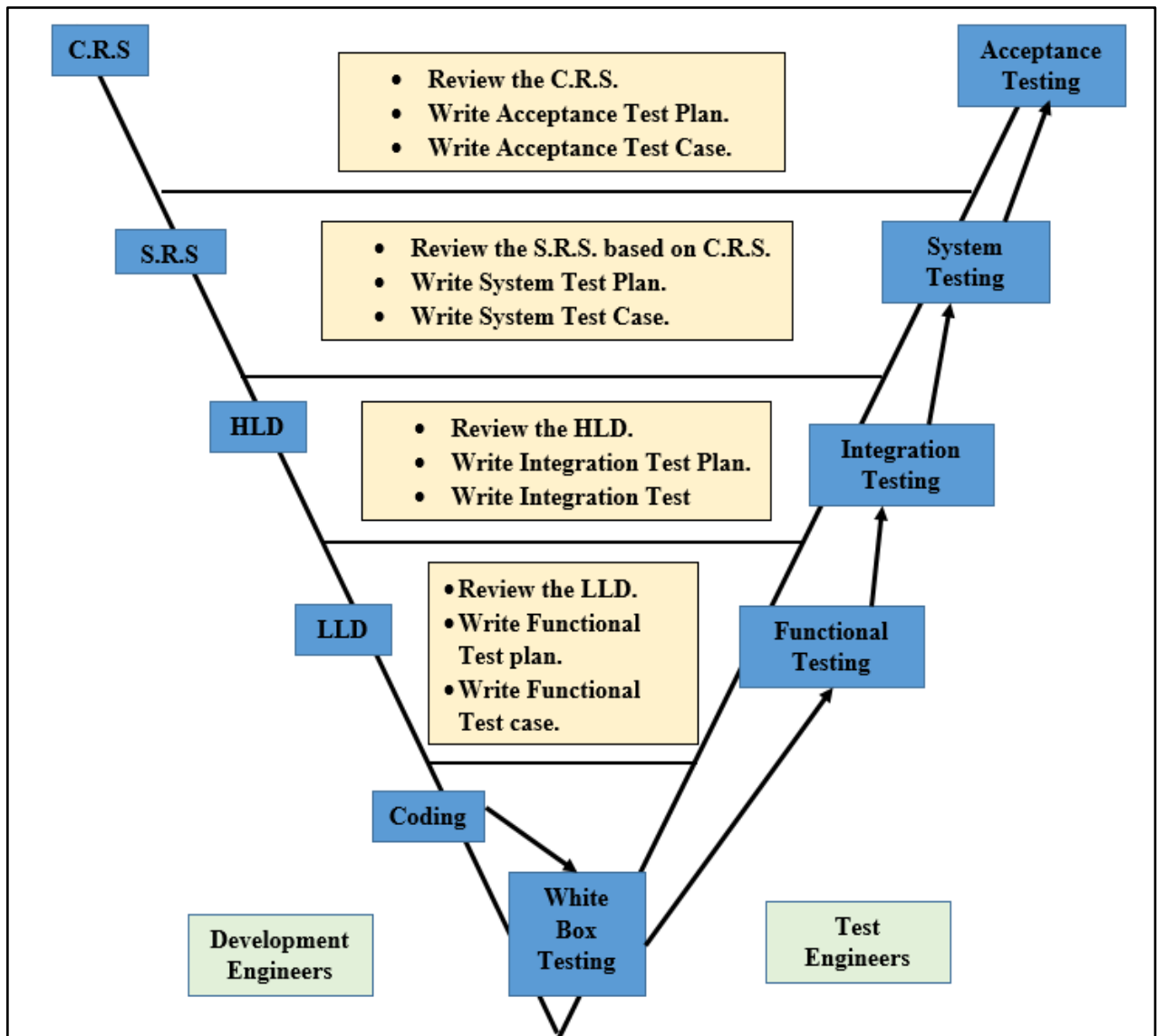
1. Each cycle looks like a waterfall model, so the same old drawbacks carry here.
2. Developers will be involved in testing.
3. Testing starts only after coding.
4. Requirement and design is not tested.

- **Applications of Spiral Model :**

1. Whenever there is dependency between the modules.
2. When requirements are given in stages.

3) V- Model:

- In order to overcome the drawbacks which were there in the waterfall and spiral model we will go for the V-model.



• Advantages of V-Model :

1. Testing starts in the very early stage of the project development that is at the requirement collection stage.
2. All the stages are well tested, because of this it avoids downward flow of defects which in turn reduces a lot of re-work.
3. Total Cost will be very less.
4. The Output is given simultaneously, because of this project gets completed very fast.

• Disadvantages of V- Model :

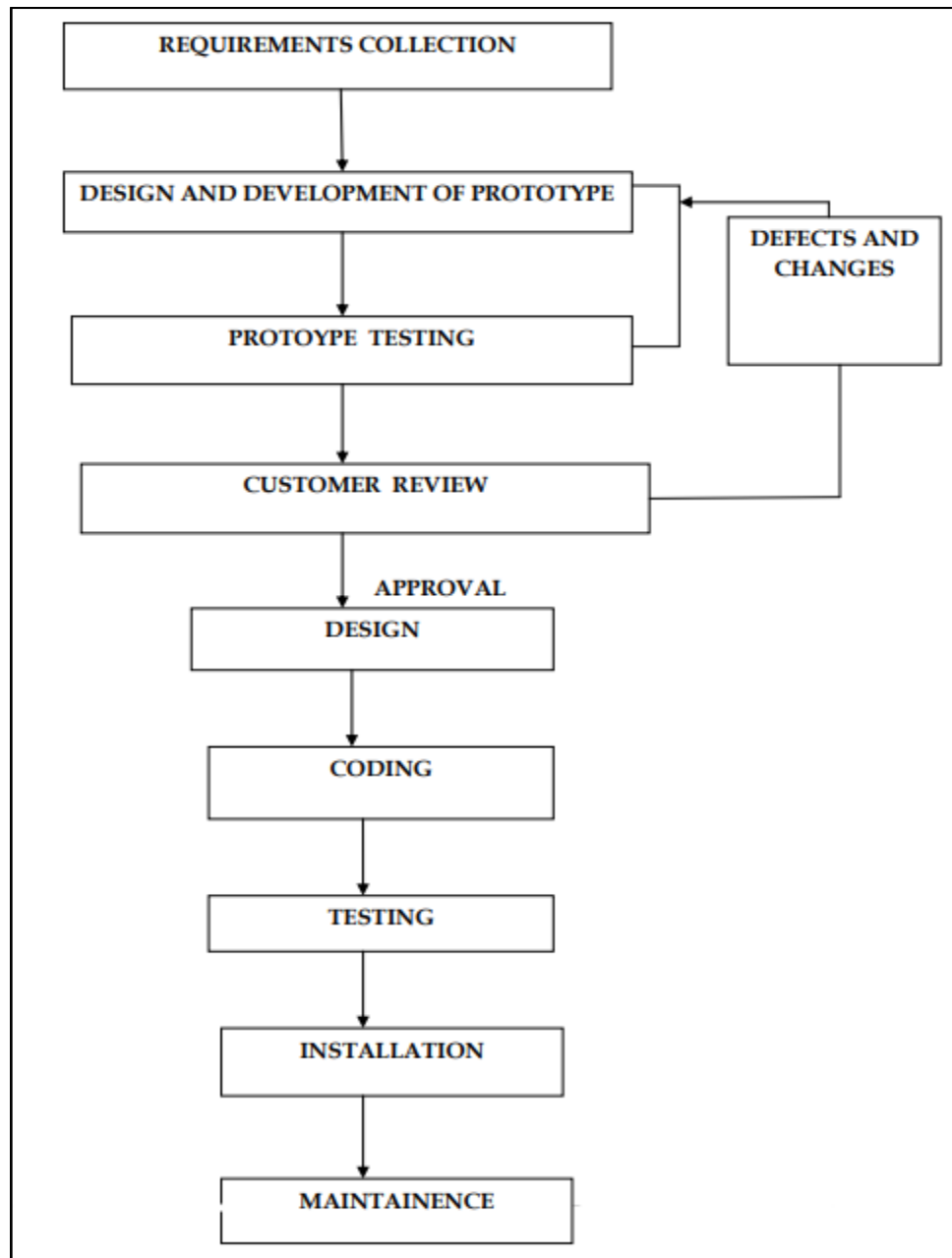
1. It involves too much Documentation work.
2. Initial Investment is high.

• Applications of V- Model :

1. Whenever we go for Long term projects.
2. Whenever we build complex and large products /applications.
3. Whenever a customer is expecting a very high quality product under pressurized high time frames.

4) Prototype Model:

- It is Dummy Software; how actual software looks like but it is not a actual software.



- **Advantages of Prototype Model :**

1. There is an improved communication between the customer, development team and testing team.
2. Initially the customer will get to know what the outcome is.
3. Initially the developers will get to know what exactly they have to develop.
4. Customer gets the opportunity to ask for the changes in the beginning itself.

- **Disadvantages of Prototype Model :**

1. Investment is needed just to build a prototype.
2. Actual development of the application starts very late, just because they are busy designing and developing prototypes.

- **Applications of Prototype Model :**

1. When a customer is not clear about explaining his own requirements.
2. When a customer is new to the software development process.
3. When developers are new to the domain.

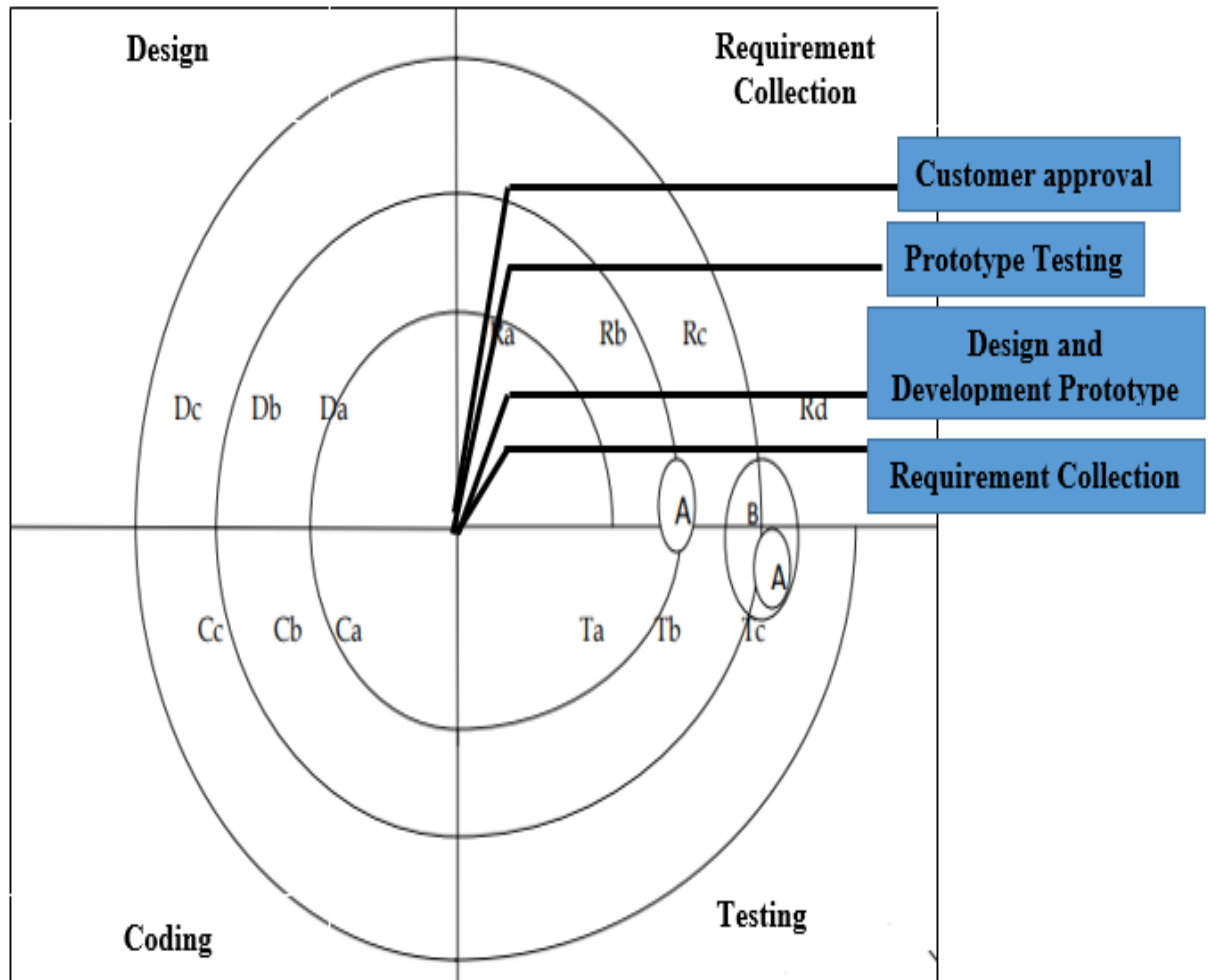
5) Customized / Derived Model :

- We take a basic model and do some changes to it, that changed model is called as customized model.

6) Hybrid Model :

- It is a combination of two models.

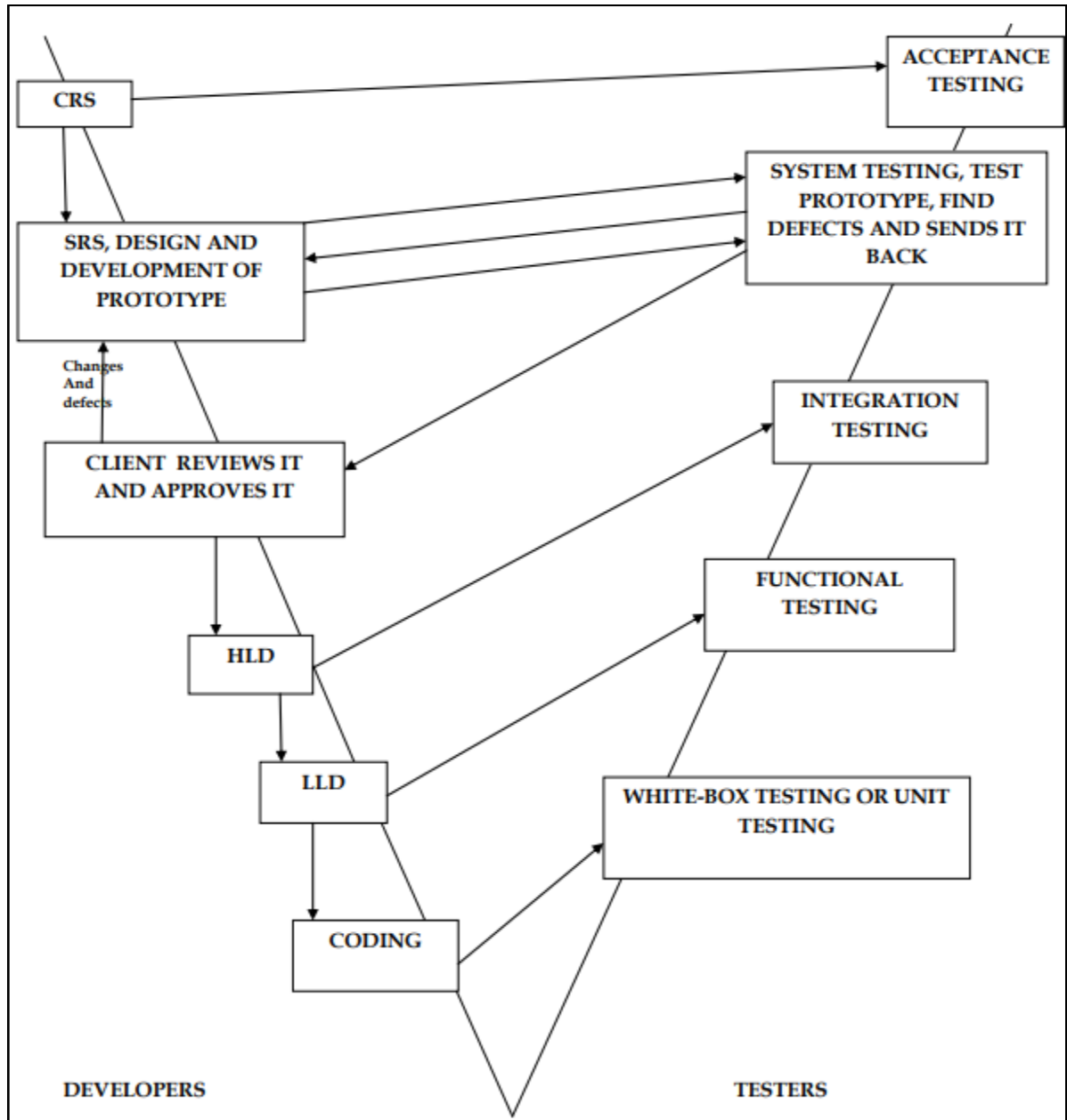
a) Spiral and Prototype model



• Applications of Prototype Model :

1. When there is dependency between the modules.
2. When a customer is not clear about the requirement.
3. When requirements are given stages.
4. When a customer is new to the process.

b) V- model and Prototype model:



- **Applications of Prototype Model :**

1. Whenever we go for Long term projects.
2. Whenever we build complex and large products /applications.
3. Whenever a customer is expecting a very high quality product under pressurized high time frames.
4. When a customer is not clear about explaining his own requirements.
5. When a customer is new to the software development process.
6. When developers are new to the domain.

WHITE BOX TESTING & BLACK BOX TESTING

1. What is Software Testing?

- It is a process of finding defects in the software is called as software testing.
- Verifying the functionality of an application against requirement specification is called as software testing.

2. Why we go for Software Testing?

- Every Software is used to support the business, If there is a defect in the software it affects the business, so before we use the application it has to be tested and all the problems should be recognized and should be fixed.
- To improve the quality of the product we go for software testing.
- To make sure that software meets the customer's requirement we go for software testing.

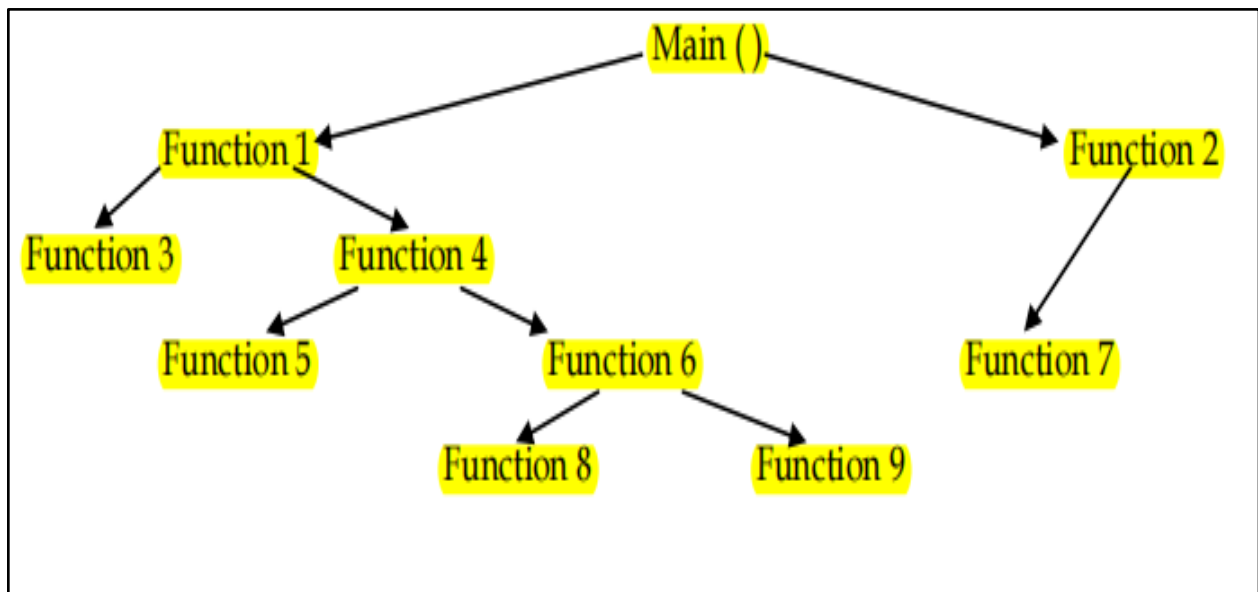
3. What are the Different types of Software Testing?

- White Box Testing (Unit Testing, Structural Testing, Glass Box Testing, Open Box Testing, Transparent Testing).
- Black Box Testing (Functional Testing, Behavioural Testing, Closed Box Testing).
- Grey Box Testing.

4. What is White Box Testing (WBT) and its types?

- Testing each and every line of the code is called as White Box Testing.
- Entire WBT is done by Developers. Developers do WBT, sends the software to Testing team.
- Types of White Box Testing are :

- 1) **Path Testing (Cyclomatic Complexity):** Writing the Flow graph and testing all the independent paths is called as Path testing.
 - Writing flow graphs means – representing the flow of the program, how each program is interlinked with one another.



- 2) **Condition Testing:** Testing all the logical conditions for both True and False Values is called as Condition Testing. i.e, we check for both 'if' and 'else' condition.

```

If( condition) - true
{
    .....
    .....
}
Else - false
{
    .....
    .....
}

```

- The program should work correctly for both conditions i.e, if condition is true, then else should be false and vice-versa.

- 3) **Loop Testing:** Testing the loops for all the cycles is called as Loop testing.

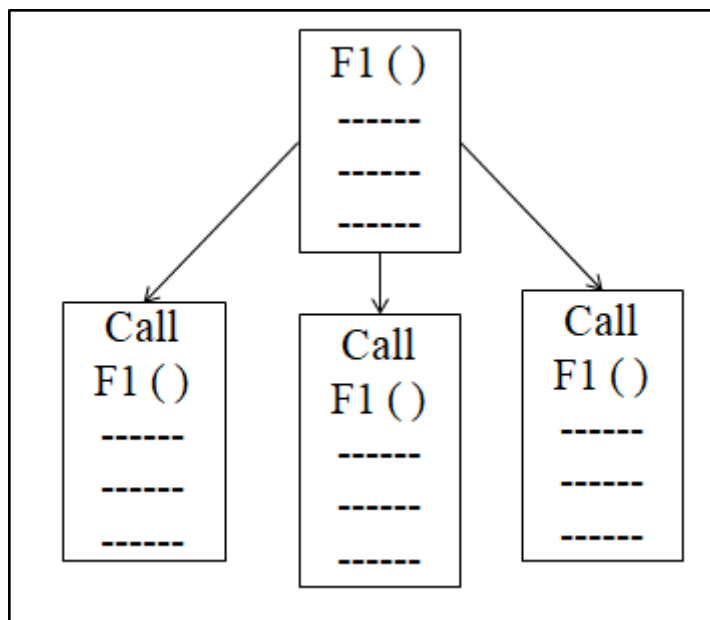
```

-----
-----
-----
n=1;
-----
-----
While (n<=10)
{ -----
  -----
  -----
  n++;
  -----
  -----
  -----
}

```

- 4) **White Box Testing from Memory Point of View:** Let's see what are all the mistakes that developer will do due to which the size of the program will increase.

- Because of Logical Mistakes:** The logic used by the programmer may vary. If one developer writes a code of 300kb file size, then another developer may write the same program using different logic and write of code of 200kb file size.
- Because of not using the Functions:** Here developer will write same repeated no. of lines of programs, instead the developers should write a single main program and this main program should be called where ever it is necessary.



- iii. **Because of not using inbuilt function:** The developer doesn't use already existing inbuilt functions and sits and writes the entire function using his logic. Thus leads to waste of time and also delays.
- Let us consider there is an already inbuilt function **sort ()**. Instead the developer sits and writes his own program **mysort ()**. This leads to waste of time and effort. Instead he could have used a single function call **sort ()**.
- iv. **Because of unused variables and unused functions:** Developers declare so many variables, functions which may never be used in any part of the program. Thus, the size of the program increases. For example,

```

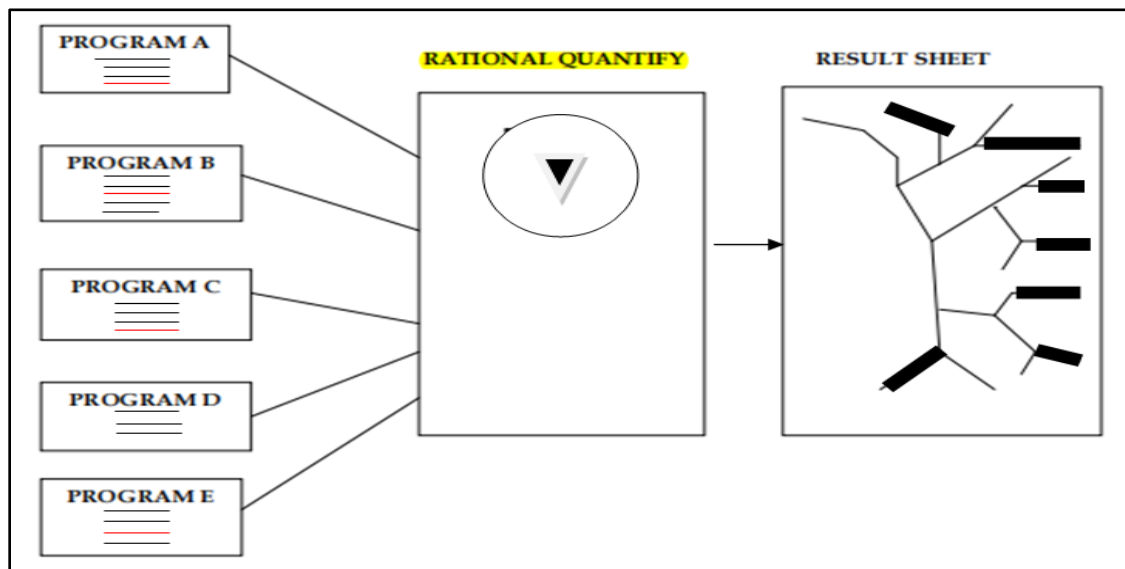
Int i = 10 ;
Int j = 15 ;
String S = "hello" ;
.....
.....
.....
.....
Int a = j ;
Create user
{
.....
.....
100 lines of code
.....
}

```

- In the above program, we can see that the integer **i** has never been called anywhere in the program and also the function **create user** has never been called anywhere in the program. Thus this leads to memory wastage.

5) **White Box testing from performance point of view:** Let's see what are all the mistakes that developer will do due to which the program is consuming more time.

- i. Because of not using better logic.
- ii. When we are using OR in the condition, put the condition in the beginning where in most of the time it gives result as True and put that condition at the end in which most of the time it give result as False.
- iii. When we are using AND in the condition, put the condition in the beginning where in most of the time it gives result as False and put that condition at the end in which most of the time it give result as True.



- As the developer is doing WBT, he sees that the program is running slow or the performance of the program is slow. The developer cannot go manually through the code and check which line of the code is slowing the program.
- We have a tool by name **Rational Quantify** to do this job automatically. As soon as all the programs are ready. This tool will go into the program and runs all the programs. The result is shown in the result sheet in the form of thick and thin lines. Thick lines indicate that the piece of code is taking longer time to run. When we double-click on the thick line, then the tool automatically takes us to the line/piece of code which is also colored differently. We can modify that code and again use rational quantify. When the sequence of lines are all thin, we know that the performance of the program has improved.

5. What is Grey Box Testing?

- Grey Box testing is combination of both White box testing and Black box Testing.

6. What is Black Box Testing (BBT) and its types?

- Verifying the functionality of an application against requirement specification is called as Black Box testing. We call this as Black Box testing because here we don't see the source code.
- Entire BBT is done by entire Testing team.
- Types of Black Box Testing are :
 - 1) Functionality Testing.
 - 2) Integration Testing.
 - 3) System Testing.

- 4) Acceptance Testing.
- 5) Usability Testing.
- 6) Performance Testing.
- 7) Compatibility Testing.
- 8) Globalisation Testing.
- 9) Smoke Testing.
- 10) Accessibility Testing.
- 11) Adhoc Testing.
- 12) Regression Testing.

7. What is Functionality Testing?

- Testing each and every component thoroughly (rigorously) against requirement specifications is known as functionality testing. It is also called as Field Level testing/ Component Testing. For example,
- This is how the requirements given by the client looks like,
 - ❖ **From Account Number (Text field):** Should accept only Alphanumeric Values and the maximum length of the text field is 8.
 - ❖ **To Account Number (Test field):** Should accept only Alphanumeric Values and the maximum length of the text field is 8.
 - ❖ **Amount (Test field):** Should accept only Numeric values.

The screenshot shows a web application for 'AMOUNT TRANSFER'. On the left is a sidebar menu with links: 'Account Balance', 'Amount Transfer', 'Loans', 'Insurance', 'Transactions', and 'Logout'. The main area contains three text input fields: 'From Account Number', 'To Account Number', and 'Amount'. Below these fields are two buttons: 'TRANSFER' and 'CANCEL'.

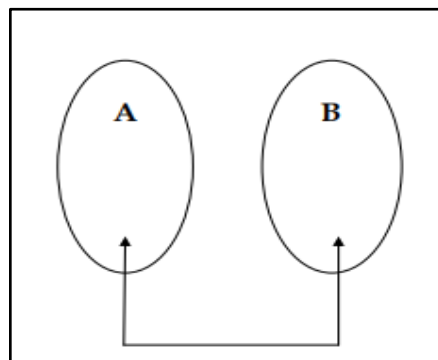
Now the test engineer does all possible tests on the 2 account numbers as:

- **From Account Number (Text field):** Should accept only Alphanumeric Values and the maximum length of the text field is 8, so the test engineer should test with the following possible data:
 - i. **By providing only Alphanumeric Values** – It should accept the values.
 - ii. **By providing only Alphabets** – It should throw the error message.
 - iii. **By providing only Numbers** – It should throw the error message.
 - iv. **By providing only Special characters** – It should throw the error message.
 - v. **By providing both Numbers and special characters** – It should throw the error message.
 - vi. **By providing Alphanumeric values and special characters** – It should throw the error message.
 - vii. **By providing less than 8 characters into the text field** – It should throw the error message.
 - viii. **By providing more than 8 characters into the text field** – It should throw the error message.

- We must always start testing the application with the valid data.
- If the application works for valid data, only then we must start testing for invalid data.
- If the application is not working for 1 of the invalid values, then we can continue testing for all the other invalid values.
- **Over Testing:** Testing the application with same scenarios in different ways or Testing the application with those scenarios that does not make sense, here we are wasting lot of time.
- **Under Testing:** Testing the application with insufficient set of scenarios.
- **Optimize Testing:** Testing the application with only those scenarios which makes sense i.e, testing for only the necessary values.
- **Positive Testing:** Testing the application with valid data.
- **Negative Testing:** Testing the application with invalid data.

8. What is Integration Testing?

- Testing the Dataflow between two modules is called as Integration Testing.



- Let us consider two modules 'A' and 'B', Data will be sent from Module 'A' to module 'B'.
- Now check whether, is there a dataflow to 'B' module.
- Check whether, module 'B' is receiving the Dataflow from module 'A'.

Amount Transfer from 'A'		Amount Received to 'B'	
<div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Account Balance</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Amount Transfer</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Loans</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Insurance</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Transactions</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Logout</div>	From Account Number <input style="width: 100px;" type="text"/> To Account Number <input style="width: 100px;" type="text"/> Amount <input style="width: 100px;" type="text"/>	<div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Account Balance</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Amount Transfer</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Loans</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Insurance</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Transactions</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 5px;">Logout</div>	From Account Number <input style="width: 100px;" type="text"/> To Account Number <input style="width: 100px;" type="text"/> Amount <input style="width: 100px;" type="text"/>
<div style="display: inline-block; border: 1px solid black; padding: 5px 15px; margin: 5px;">TRANSFER</div> <div style="display: inline-block; border: 1px solid black; padding: 5px 15px; margin: 5px 10px;">CANCEL</div>		<div style="display: inline-block; border: 1px solid black; padding: 5px 15px; margin: 5px;">TRANSFER</div> <div style="display: inline-block; border: 1px solid black; padding: 5px 15px; margin: 5px 10px;">CANCEL</div>	

- Now let us consider the example of banking s/w as shown in the figure above (amount transfer)

Scenario 1 – Login as A to Amount Transfer – send 100rs amount – message should be displayed saying 'amount transfer successful' – now logout as A and login as B – go to amount balance and check balance – balance is increased by 100rs – thus integration test is successful.

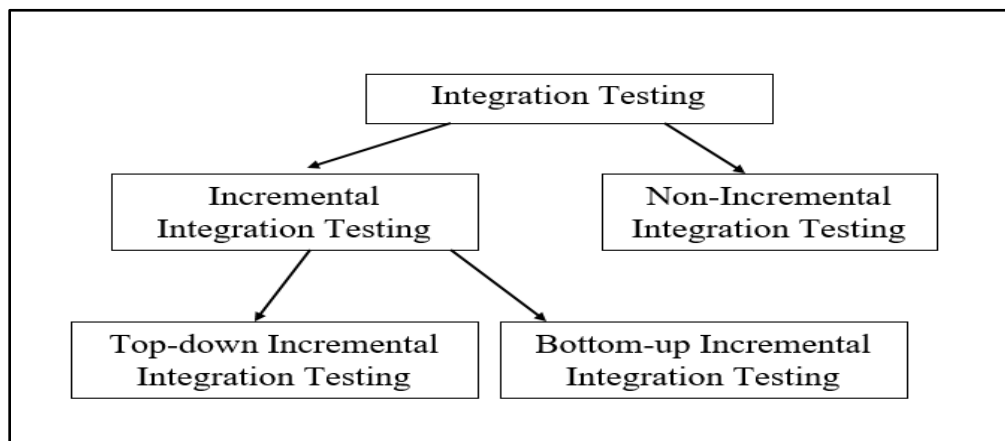
Scenario 2 – Also we check if amount balance has decreased by 100rs in A.

Scenario 3 – Click on transactions – in A and B, message should be displayed regarding the data and time of amount transfer.

- **Thus in Integration Testing, we must remember the following points,**

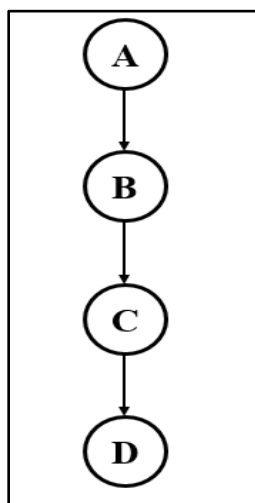
1. Understand the application thoroughly i.e, understand how each and every feature works. Also understand how each and every feature are related or linked to each other.
2. Identify all possible scenarios.
3. Prioritize all the scenarios for execution.
4. Test all the scenarios.
5. If you find defects, communicate defect report to developers.
6. Do positive and negative integration testing:
 - **Positive** – if there is total balance of 10,000 – send 1000rs and see if amount transfer works fine – if it does, then test is pass.
 - **Negative** – if there is total balance of 10,000 – send 15000rs and see if amount transfer happens – if it doesn't happen, test is pass – if it happens, then there is a bug and send it to development team for repairing defects.

- Integration Testing is classified into two types:

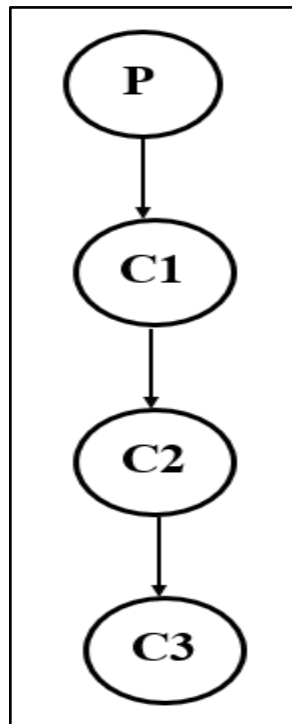


1) **Incremental Integration Testing:** Incrementally add the modules and test the data flow between the modules.

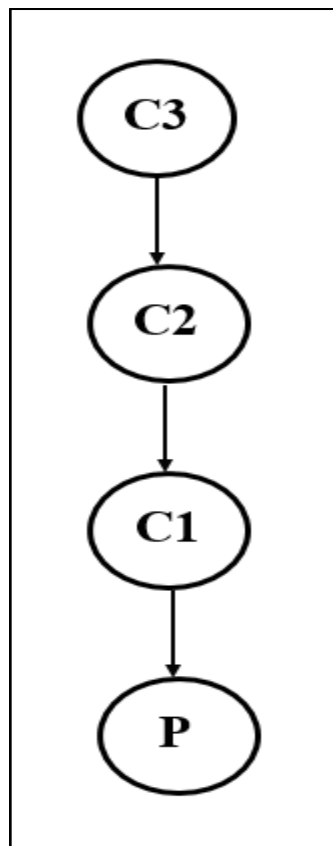
- Take two modules. Check if data flow between the two is working fine. If it is, then add one more module and test again. Continue the same, incrementally add the modules and test the data flow between the modules.



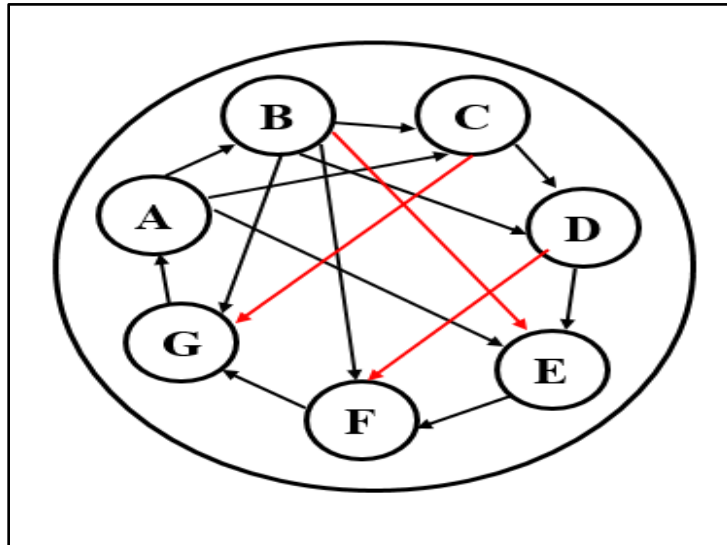
- There are two types of Incremental Integration Testing :
 - i. **Top-down Incremental Integration Testing:** Incrementally add the modules and test the data flow between the modules. Make sure that the module that we are adding is child of previous one. Child3 is child of child2 and so on.



- ii. **Bottom-Up Incremental Integration Testing:** Incrementally add the modules and test the data flow between modules. Make sure that the module you are adding is the parent of the previous one.

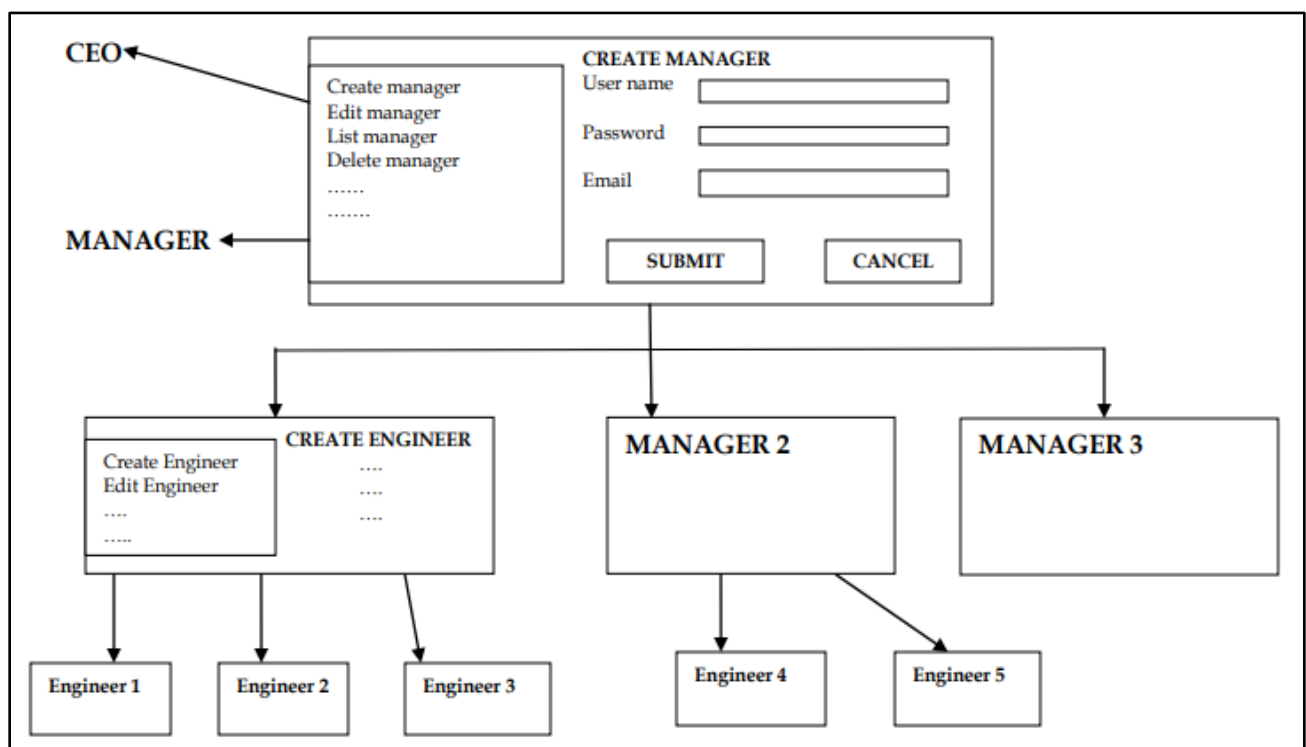


- 2) **Non- Incremental Integration testing:** Combine all the modules at a shot and start testing the data flow between the modules, this type of testing is called Non-Incremental Integration testing.



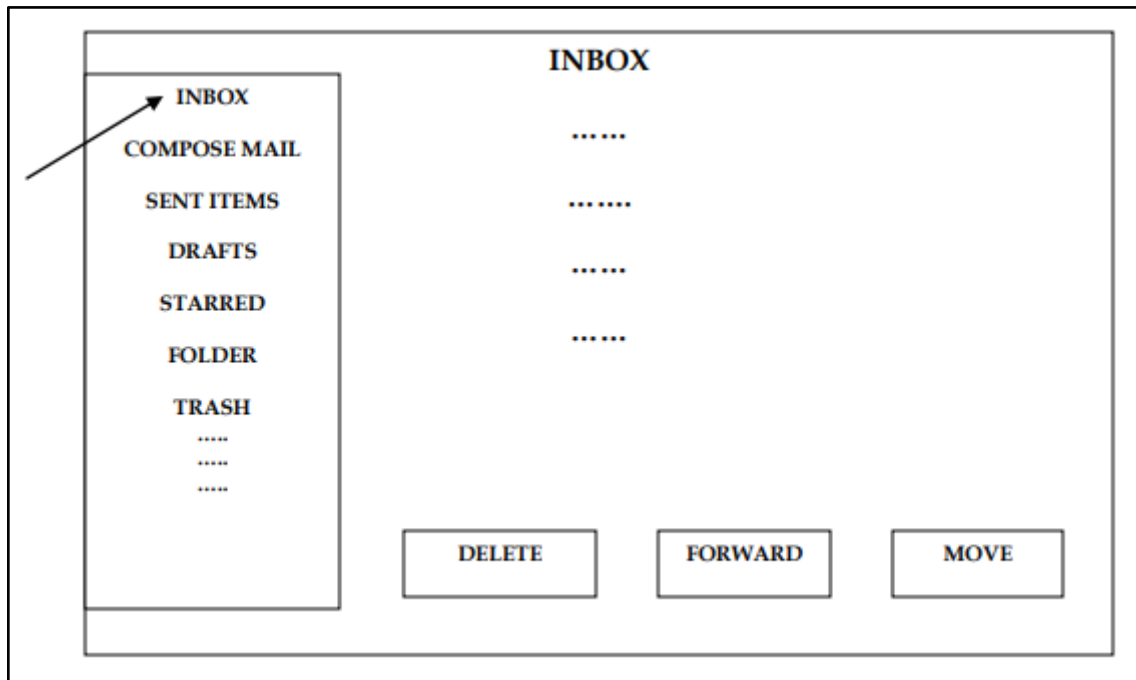
- Here we can't identify which is Parent module and which is Child module.
- Here we may miss the integration testing between some modules.

Example of Incremental Integration Testing:



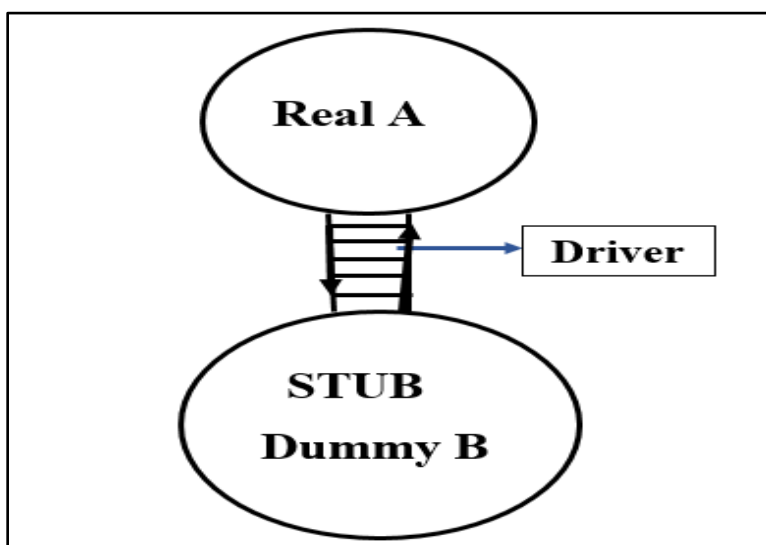
- In the above example. The development team develops the s/w and send it to the CEO of the testing team. The CEO then logs onto the s/w and creates the username and password and send a mail to a manager and tells him to start testing the s/w. The manager then edits the username and password and creates a username and password and send it to the engineer for testing. This hierarchy from CEO to Testing Engineer is top-down incremental integration testing. Similarly, the testing engineer once he finishes testing sends a report to the manager, who then sends a report to the CEO. This is known as bottom-up incremental integration testing.

- **Example for Non-Incremental Integration testing:** The example displays a homepage of a Gmail inbox. When we click on an inbox link, we are transferred to the inbox page. Here we have to do non-incremental integration testing because there is no parent and child process here.



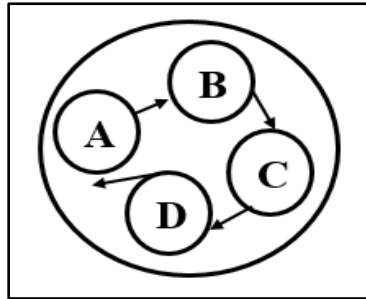
❖ **When one module is ready and another module is not ready, in such a case how will you do Integration testing?**

- ★ **Stubs:** Stub is a dummy module which just receives data and generates a whole lot of expected data, but it behaves like a real module. When a data is sent from real module A to stub B, then B just accepts the data without validating and verifying the data and it generates expected results for the given data.
- ★ **Driver:** Driver is one which sets up the test environment and takes care of communications, analyses results and sends the report.

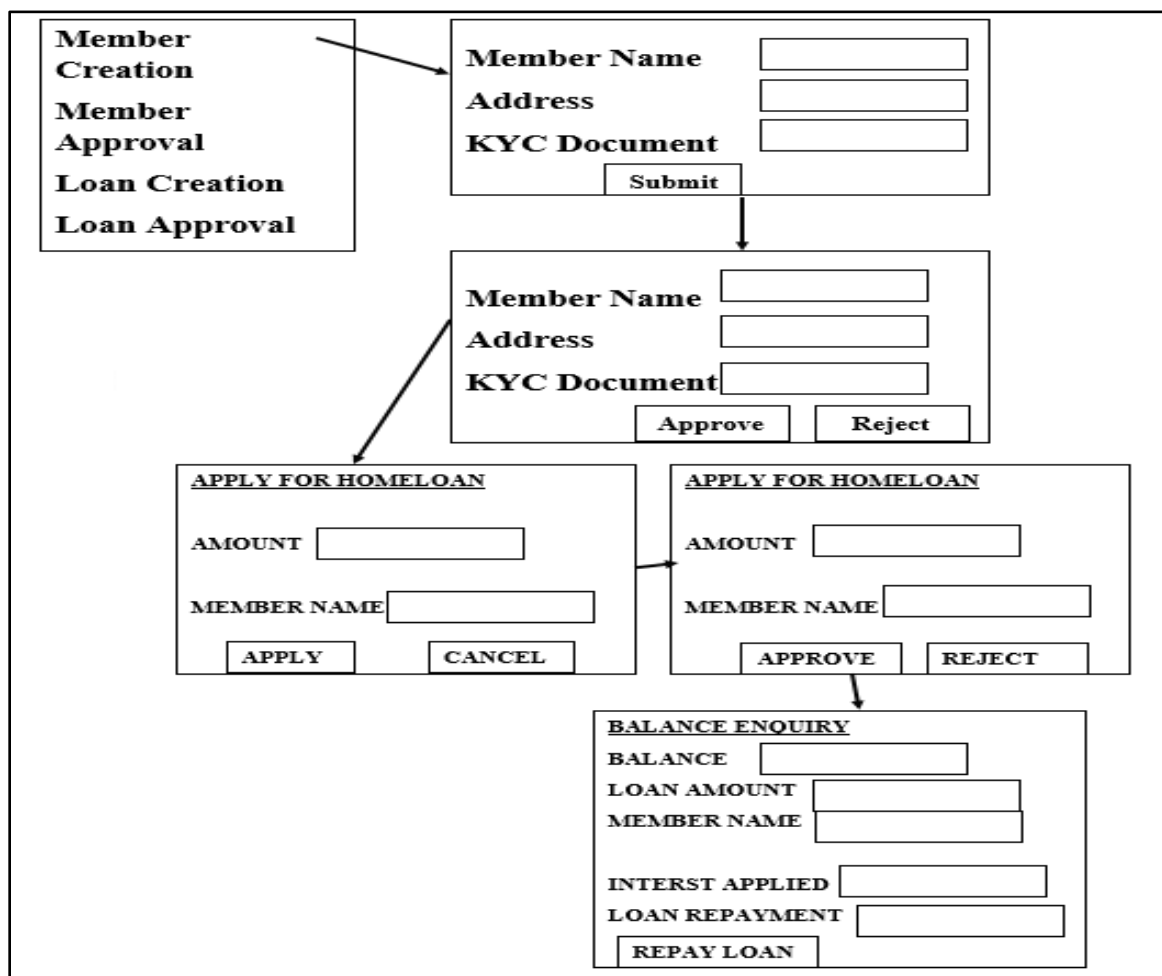


9. What is System Testing?

- It is end-to-end testing wherein the testing environment is similar to the production environment.
- **End - to-End Testing:** Take End-to-End business scenarios and check whether software is capable of handling it.



Let us consider an example to explain System Testing:



1. Login as A - Apply for Member Creation.
2. Login as Manager - Approve the Member.
3. Login as A - Apply for Home Loan of 2,00,000/-.
4. Login as Manager - Approve the Loan.
5. Run the EOD (Change the server date to next 30 days).
6. Login as A - Check the balance 2,00,000/- should be deposited.
7. As per frequency (Monthly) the repayment schedule for 10 months will be generated and the the amount will be deducted (2,00,000 >> 2% = 20,000 per month + 2%) should be paid to bank.
8. Login as Manager - Check whether the Loan amount for the particular month is accrued as per the schedule.

When we will do System Testing?

- When minimum bunch of features are ready.
- When all the basic features are stable.
- If the environment similar to production environment is available.

When do you release the product to the production?

- When all the features requested by the customer are ready.
- When there are no blocker or critical bugs left in the product.
- When there are few minor bugs left out, which are acceptable by customer.
- **What is One Test Cycle:** It is time spent or the effort that you have put to start and finish the complete product testing is called as one Test cycle.

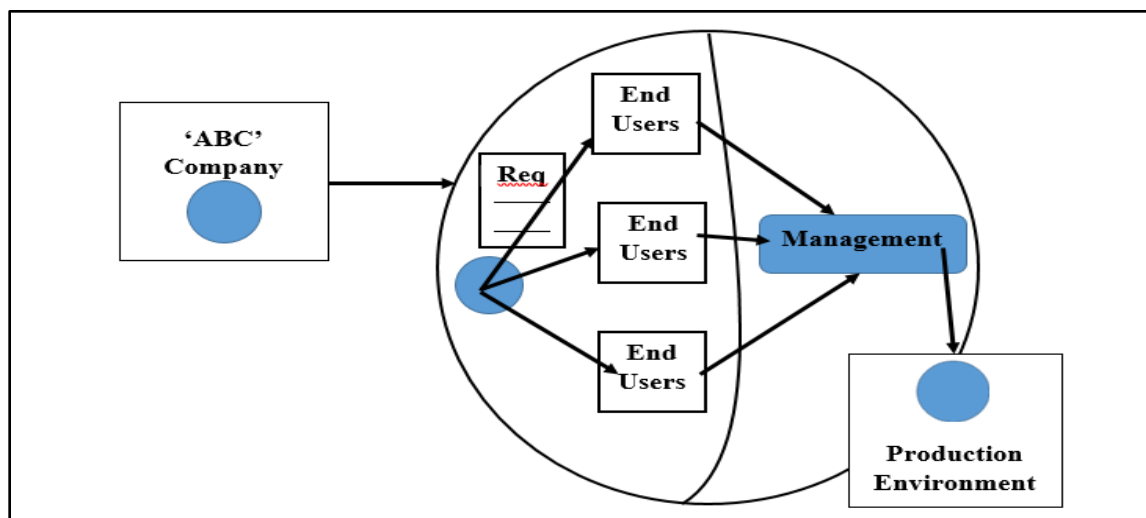
OR

One Test cycle duration depends on the size of application, depends on the complexity of the application and number of test engineers.

- **What is Build:** All the developers will write program, all the programs are compiled and the format will be in Binary, All the Binary's will be compressed this compressed file is called as Build.
- **What is Patch:** Patch is a small build or software which contains modified, new and the record of deleted programs.

10. What is Acceptance Testing?

- Acceptance testing is done by end users. Here, they use the software for their business for a particular period of time and check whether the software can handle all kinds of real-time business scenarios.



OR

- It is end to end testing done by engineers sitting at customer place, where in they take end to end real time business scenarios and check whether software is capable of handling it or not.

OR

- It is end to end testing done by our own engineers sitting at customer's place, where in they refer to user scenarios given by customer and check whether software is capable of handling it or not.

OR

- It is end to end testing done by our own engineers sitting at software company, by referring to user scenarios given by customers and check whether software is capable of handling the real time business scenarios.

Why we do acceptance testing?

- Chances are there under business pressure, software company might push the software with lot of critical bugs, to find it customer does Acceptance testing.
- If there is a critical bug and if customer uses the software for business, then the customer will undergo severe loss, to avoid this they go for acceptance testing.
- Chances are there developers would have misunderstood the requirements and developed the wrong feature, to find that we go for acceptance testing.

11. What is Usability Testing?

- It is also called as GUI testing (Graphical User Interface testing) / Cosmetic testing.
- Testing the User friendliness of an application is called as Usability testing.

How to do Usability Testing?

- Here we will check the Look and Feel of the application is good or not.
- Here we will check whether application is simple to understand or not.
- Important features or frequently used features must be given to the user within 3clicks.
- Important features or frequently used features should be easily accessible.
- Important features or frequently used features should be present at left navigation or top navigation bar.
- To do simple activities in the application, the application should take less time.

For What kind of applications we should do Usability Testing?

- Any application which is used by Multiple users.
- Any application which generates lot of Revenue.
- We have to perform usability testing for those kind of applications, where in we do not provide any training to the end users.

When to do Usability testing?

- Some projects they do usability testing when the project becomes functionally stable.
- Certain projects they do usability testing in the beginning of SDLC itself.

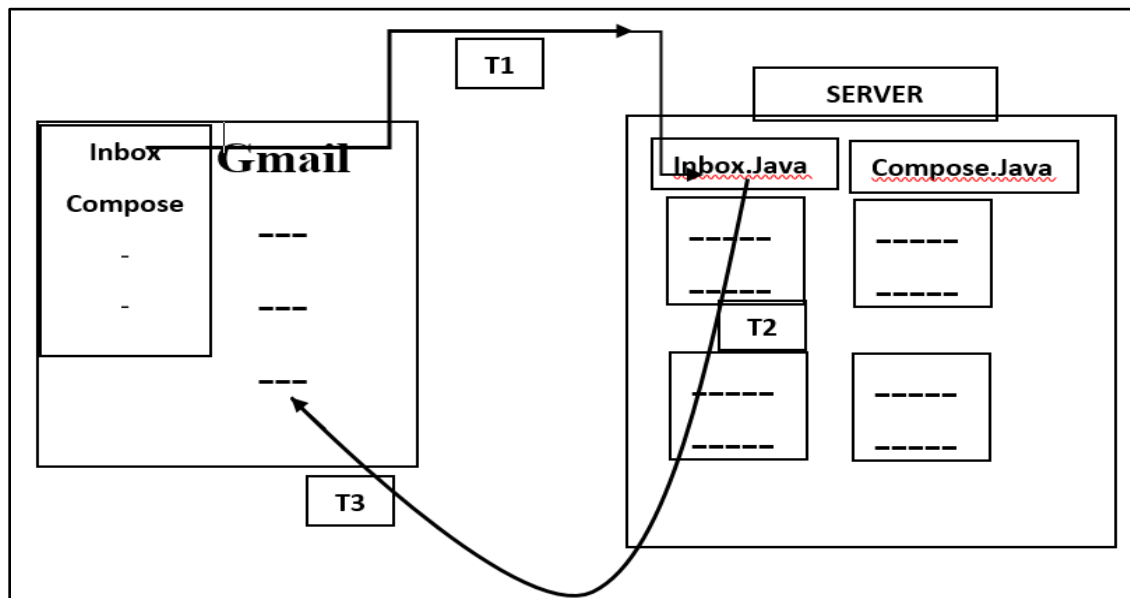
12. What is Performance Testing?

- Testing the Stability and Response time of an application by applying Load is called as Performance Testing.

Response Time: It is the Total time taken to send the Request(T1), time taken to execute the program(T2), time taken to receive the Response(T3). **RT = T1+T2+T3**

Load: It is the Designed Number of Users (Number of Requests).

Stability: It is the ability to withstand the designed number of users.

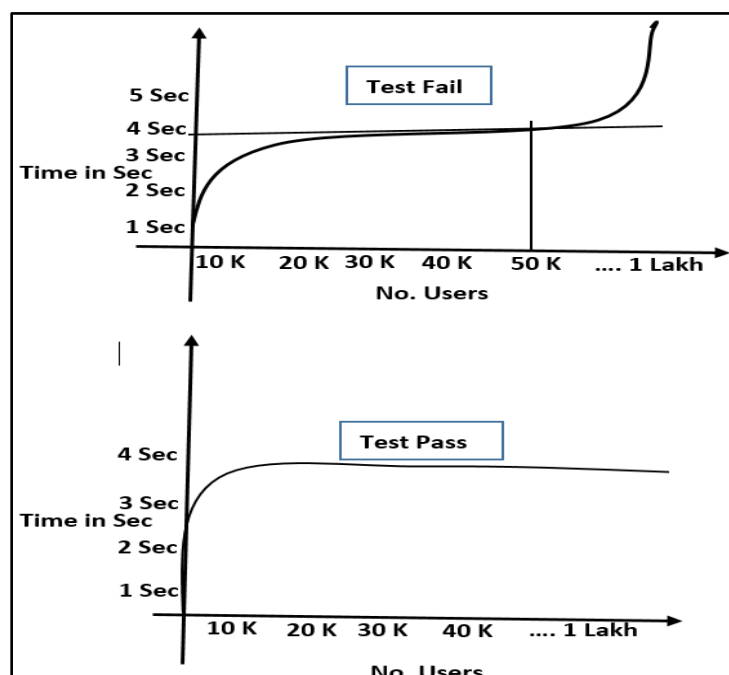
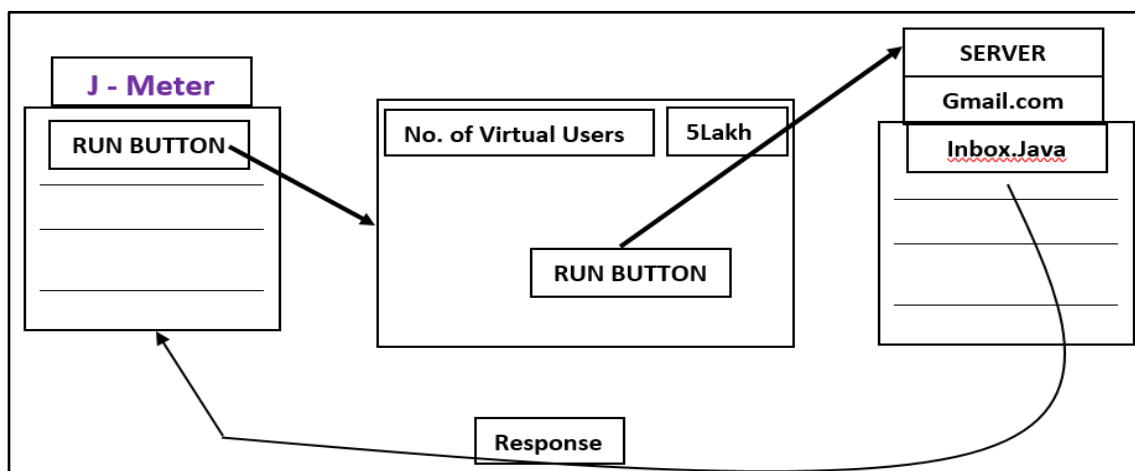


Different types of Performance Tools?

- J- Meter.
- Neo- Load.
- Load- Runner.

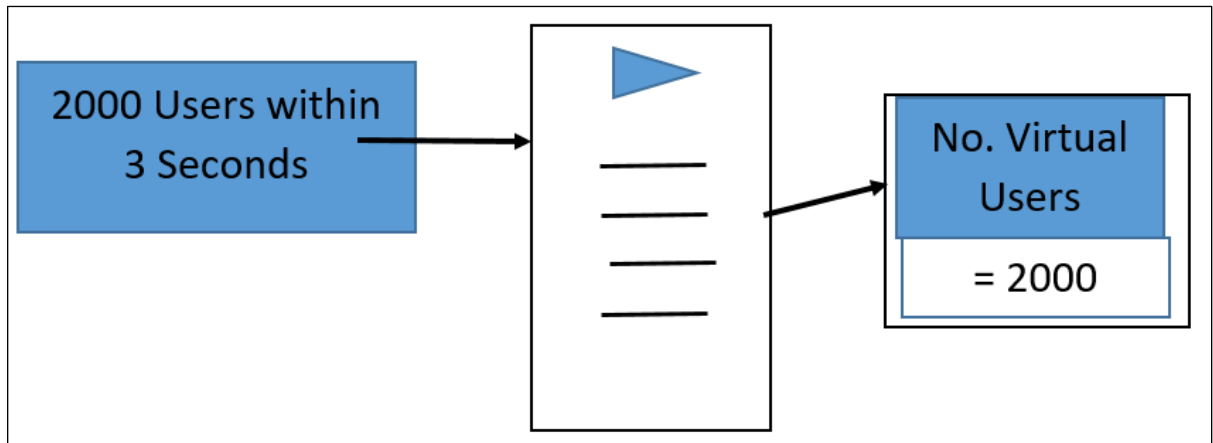
How to do Performance Testing Using Performance Testing Tools?

- Take Performance Tool (J-Meter) and write scripts and click on “RUN” button.
- The Tool will ask for Number of Virtual Users, for example let us take 5Lakh users and Click on RUN button.
- Now this 5Lakh request will be fired to the server and there is a heavy load on the server.
- Now the server will execute the program and send response to the Performance Testing Tool (J-Meter).
- The tool will analyse the response and gives us the result in the form of Graph.
- Now the Test Engineer will analyse the graph and gives the Result that whether test is Pass/Fail.
- Suppose if the test is Fail, then the Test Engineer will communicate the defects to the Development Engineers.
- Now the Development Engineers will go to the code and fix the defects in order to improve the Performance and will gives new build to the Testing Team.
- Now the Test engineer again will test the software with the help of Performance tool and the same process repeats until the Test is Pass.

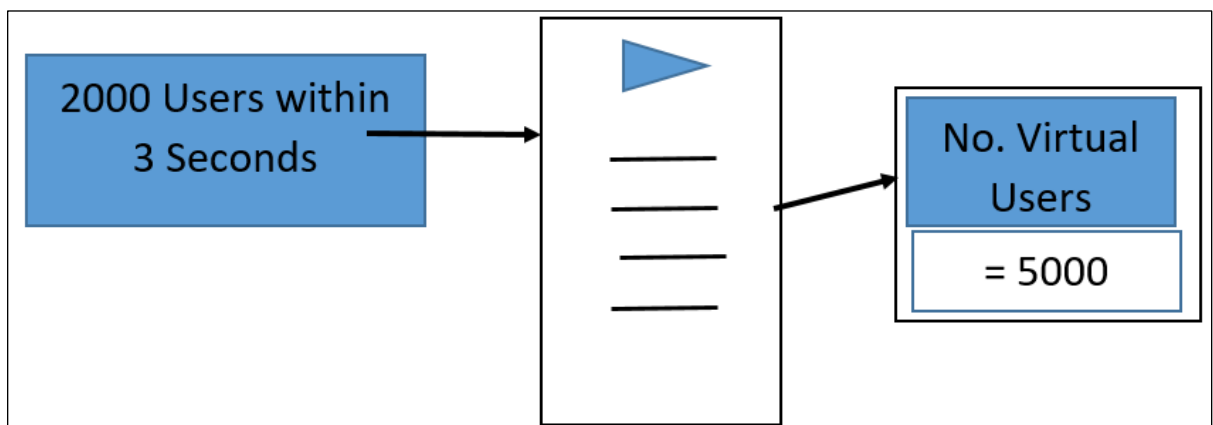


Different types of Performance Testing:

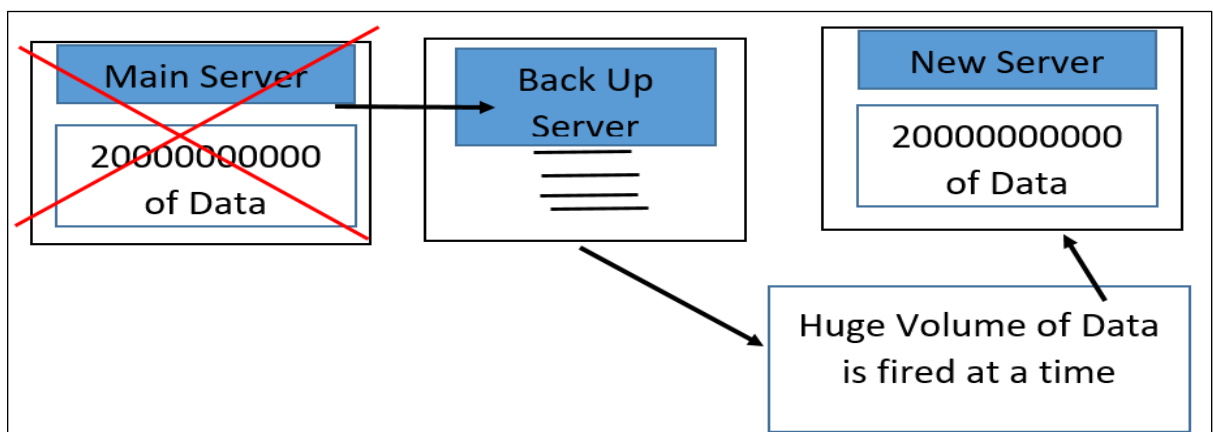
- Load Testing.
 - Stress Testing.
 - Volume Testing.
 - Soak Testing.
- **Load Testing:** Testing the Stability and Response time of an application by applying Load which is less than or equal to the Designed number of users is called as Load Testing.



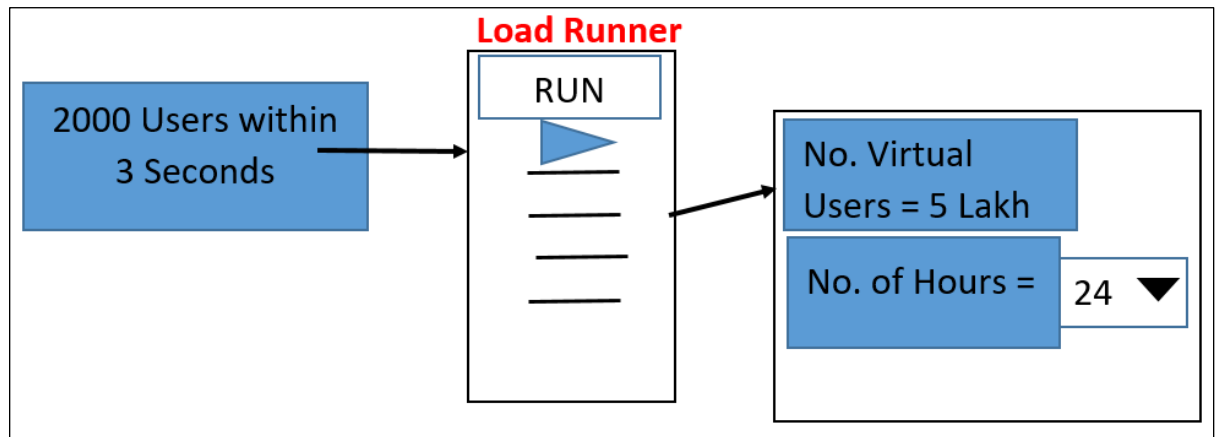
- **Stress Testing:** Testing the Stability and Response time of an application by applying Load which is more than the Designed number of users is called as Load Testing.



- **Volume Testing:** Testing the Stability and Response time of an application by transferring huge volume of data is called as Volume Testing.



- **Soak Testing:** Testing the Stability and Response time of an application by applying Load continuously over a particular period of time is called as Soak Testing.



For what kind of applications we do Performance Testing?

- Any application which is used by Multiple Users.
- Any application which generates lot of Revenue.

When we do Performance Testing?

- For certain Projects we do performance testing when the product becomes Functionally stable.
- Certain projects they do performance testing in the beginning of SDLC itself.

13. What is Compatibility Testing?

- Testing the software or application in different hardware and software environment is called as compatibility Testing.

Why we do Compatibility Testing?

- Chances are there Developer might develop the software (application) in one platform (windows 10) and Test Engineer might test the software in the same platform (windows 10) and when it is released to the production, end users might use the application in different platforms i.e. (windows 7 or windows 8). Software which works in one platform may not work in other platform, so to avoid this we go for compatibility testing.
- We do compatibility testing to check whether features are working consistently in all the platforms.
- Chances are there that developers would have written common code and they claim that it works in all the platforms.
- Different hardware and software renders GUI in different ways, we may have to test whether the application is rendered properly for different combination of hardware and software environment.

When we do Compatibility Testing?

- When the product becomes functionally stable in the base platform, then we start testing it in other platforms.

What kind of Bugs do we get while we do Compatibility Testing?

- Alignment Issue.
- Scattered Issues.
- Object overlapping.
- Scroll bar Issues.

14) What is Globalization Testing?

- Developing the software for multiple or different languages is called Globalization and testing the software which is developed for different Languages is called as Globalization Testing.
- There are two types of Globalization Testing:
 - ❖ Internationalisation Testing (I18N) Testing.
 - ❖ Localization (L10N) Testing.

❖ Internationalisation Testing (I18N) Testing:

- Testing the application or software which is developed for multiple languages is called Internalisation Testing.
- Here we check whether the content is in the right language or not.
- Right content is displayed in right place or not.
- We will check whether features are broken if the language changes.

✓ How to do I18N Testing?

1. Go the Language (French) Property file.
2. Add “Prefix” and “Suffix” to the content and save the property file.
3. Open the Application, select the language as (French) and the corresponding application will be displayed in the same language.
4. We will check for the “Prefix”, if Prefix is correct then the “Content” is in the right language, if the “Suffix” is correct, then the right content is displayed in right place.

- * Test Engineers for their Understanding purpose on in order to test the software, they give Prefix and Suffix for different languages and contents and the process is called “**Pseudo Translation**”.

✓ What kind of defects we can expect while doing I18N testing?

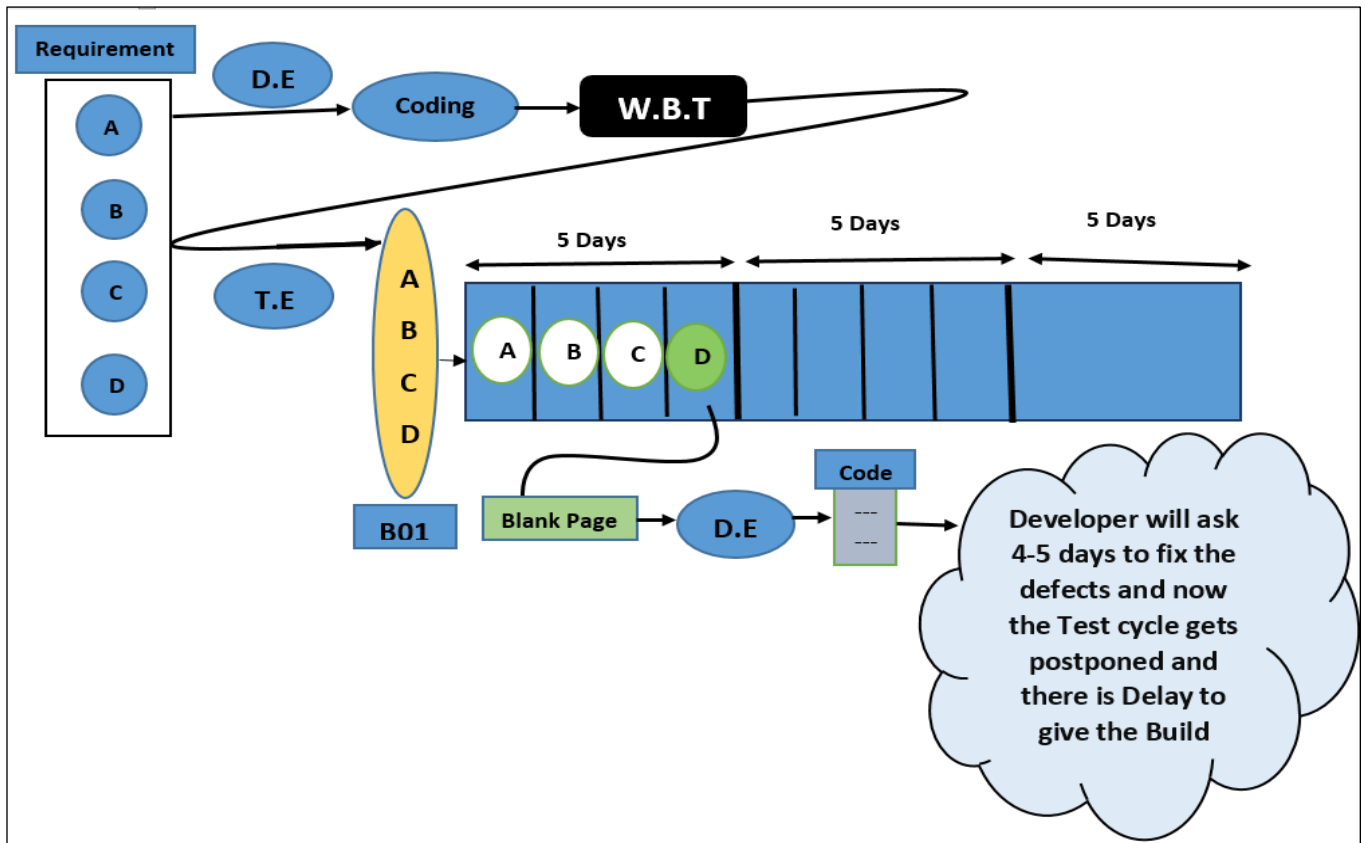
- Chances are there that the right language may not be displayed.
- Chances are there that the right content might not be displayed in the right place.
- Tooltip Defects: when point curser on the image, a rectangular box will be displayed which explains about the image.

❖ Localization Testing (L10N) Testing:

- Testing the software to check whether software is Developed according to “Country standards” or “Country Culture” is called as Localization Testing.
- **Format Testing:** Here we check whether Currency format, Pin Code is displayed as per the country standards or not.
- Here we will check whether data format is displayed as per the country Standards or not. E.g.: India- DD-MM-YYYY, US- MM-DD-YYYY, German- YYYY-MM-DD.
- Here we will check whether Colour of the image is displayed as per country standards or not.

15) What is Smoke Testing?

- Testing the Basic and critical features of the application before doing thorough testing or rigorous testing is called as Smoke Testing.



- According to above diagram, we have the requirements for A, B, C, D modules.
- Developers will write the code and do the White Box Testing (W.B.T) and will give the build to the Test Engineers.
- Test Engineers started to test the modules A, B, C for first 3 Days and now on the 4th Day when Test Engineer click on module 'D' it is showing "Blank Page" (Blank Page Means Blocker i.e., it is blocking the Test Engineer to test the application.)
- Now the Test Engineer will immediately communicate the defect to the Development team and now the developer will analyse the defect and will say that want at least 3 to 4 days to fix this defect, now because of this defect the entire testing is lagging and the Test Cycle will get postponed and release will be delayed to the customer.
- Now customer will undergo loss and will put the penalty for the software company.
- So, in order to overcome this problem, in every company as soon as they get the build or Software they should do "SMOKE" Testing.

Advantages of Smoke Testing:

- Test Engineer can find all the blocker defects in the early stage itself.
- Developer will get sufficient time to fix the defects.
- Test cycle will not be postponed and release will not be delayed.

✓ **Note:** Here we do only Positive Testing.

Why to do Smoke Testing?

- To check whether the product is testable or not, it means in the beginning only if the test engineer finds too many defects then product is not testable and the Test Engineer can stop the testing and can spend the time in identifying some scenarios.
- We do Smoke testing in the beginning only because if we find Blocker defects and send it to developers, the developers will get sufficient time to fix the defects.
- We do this to ensure that product is installed properly or not.
- It is like health check of the product, here we will check whether we have received broken build from the developers or not.

When to do Smoke Testing?

- As soon as you get new build from development team, test engineers should start with Smoke testing because test engineers are getting new build means, either developers would have added or modified or removed some features or they would have fixed the bug also, so this might affect basic and critical features of the application, so to avoid this test engineer will do smoke testing.
- Customer will do Smoke testing before doing Acceptance testing (AT) or User Acceptance Testing (UAT), because to check whether product is properly installed or not and also to check whether, he has received the product properly or not.
- One who install the product in the server, should do Smoke testing to check whether the product is installed properly or not.
- Developers before they give build to testing team, they should do Smoke testing because, if there are too many bugs then they need not give product to testing team.

Types of Smoke Testing:

- * **Formal Smoke Testing:** Developer will send build to test lead and now test lead will assign features to the test engineers and ask them to do smoke testing and every test engineers will do smoke testing for their respective assigned features. They will prepare smoke test report and send it to test lead. The test lead will consolidate smoke test reports and will send it to Development Lead. This Entire process is done formally through email or meetings, so it is called as Formal Smoke Testing.
- * **Informal Smoke Testing:** Developer will send build to test lead and now test lead will assign features to the test engineers and test lead will not ask the test engineers to prepare the smoke testing reports and test engineers will do the smoke testing by themselves. This type of testing is called as Informal Smoke testing.

Difference between Smoke Testing and Sanity Testing:

SMOKE TESTING	SANITY TESTING
Smoke Testing is Positive Testing.	Sanity Testing is Positive and Negative Testing.
Here we write and Document the Scenarios.	Here we don't write and Document the Scenarios.
Here we go for Automation.	Here we don't for Automation.
Smoke testing is done by both Developers and Test Engineers.	Sanity testing is done by only Test engineers.
Smoke Testing is Shallow and wide testing.	Sanity testing is deep and Narrow testing.

16) What is Accessibility Testing?

- Accessibility Testing is a part of usability testing where in the application is tested for differently abled end users such as users with hearing disabilities, old age, colour blindness etc.
- We follow 4 principles:
 - ✓ **Perceivable:** Users should perceive all the contents that are displayed on the screen.
 - ✓ **Operable:** Navigation and User interface should be operable to the user using Voice assistance.
 - ✓ **Understandable:** The information and operation must be understandable to the users.
 - ✓ **Robust:** The content should be robust enough to interpret with assistive technology.
- **Compliance Levels of Accessibility Testing:**
 - ✓ **A** – Must Have (Minimum Requirement).
 - ✓ **AA** – Should Have (Asks for Audio and Video caption).
 - ✓ **AAA** - Good to Have (Indicates the highest level).

17) What is Adhoc Testing?

- Testing the software or application randomly without looking into the requirement is called as Adhoc testing.

Why to do Adhoc Testing?

- When the product is launched end user might use the application randomly because of that he might get defects. To prevent such issues TE only test the application randomly and find bugs.
- If you follow the requirement and test the application number of bugs you going to catch is very less. So do not follow the requirement come up with creative scenarios which are out of requirement and test the application.
- Since customers and end-users are not following requirements test engineers should do Adhoc testing.
- We do this to increase the bug count.
- We do this to improve the test coverage.

When to do Adhoc Testing?

- When we do smoke testing we are not supposed to do Adhoc testing because Smoke testing is positive testing and adhoc testing is negative testing.
- While doing FT, IT, ST either in between or at the end, if you have some time, we do Adhoc testing.
- Once the complete software is tested as per the requirement and if the product is stable then we can do Adhoc testing.
- Once the software is tested for 10 to 15 cycles as per the requirement and if the product is stable then we can do Adhoc testing.
- While doing FT/IT/ST, if you come with good adhoc scenarios test for the adhoc scenarios but do not spend more time in doing adhoc scenarios switch back to normal testing.

Types of Adhoc Testing?

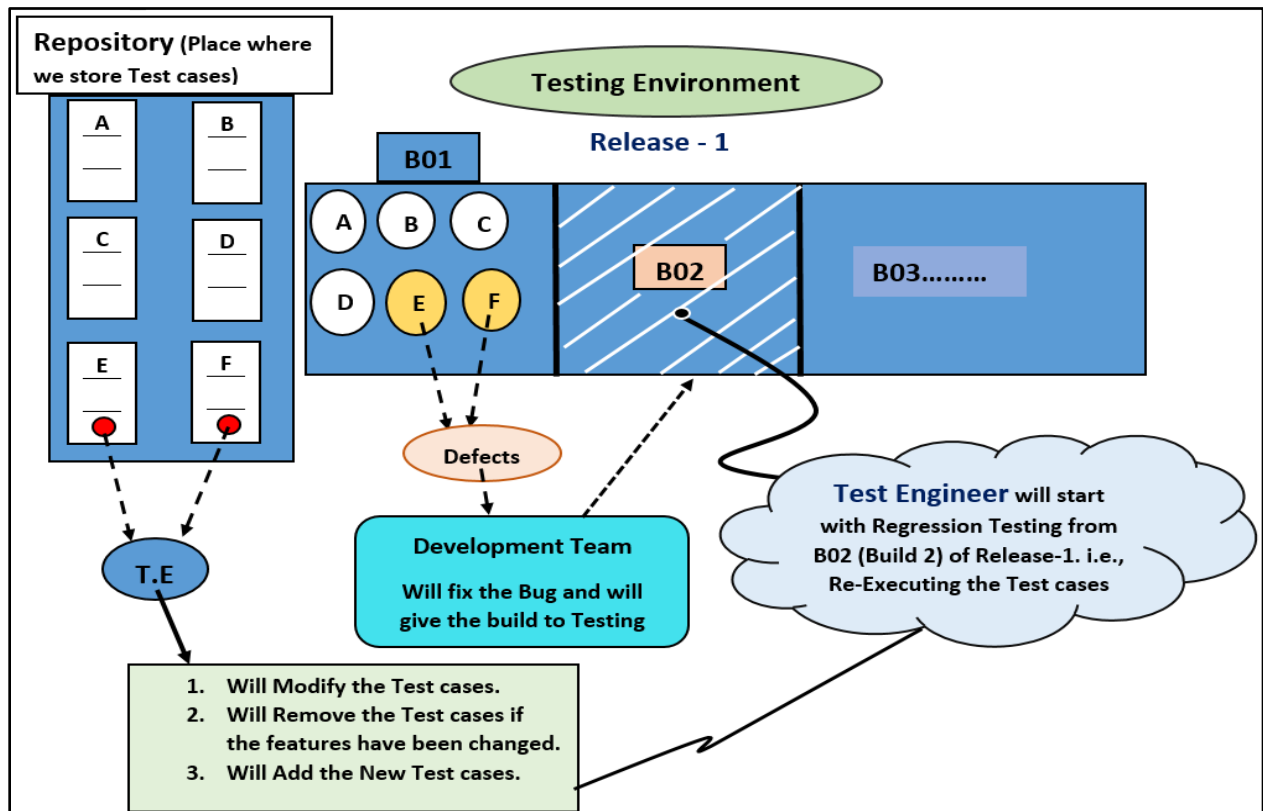
- **Monkey Testing:** Here test engineer will test the application(Software) like a monkey without applying any logic.
- **Buddy Testing:** Here the test engineer will sit with Developer and come up with creative scenario and will do Adhoc testing.
- **Pair Testing:** Here test engineer will sit with another test engineer and comes with creative scenario and will do adhoc testing.

18) What is Regression Testing?

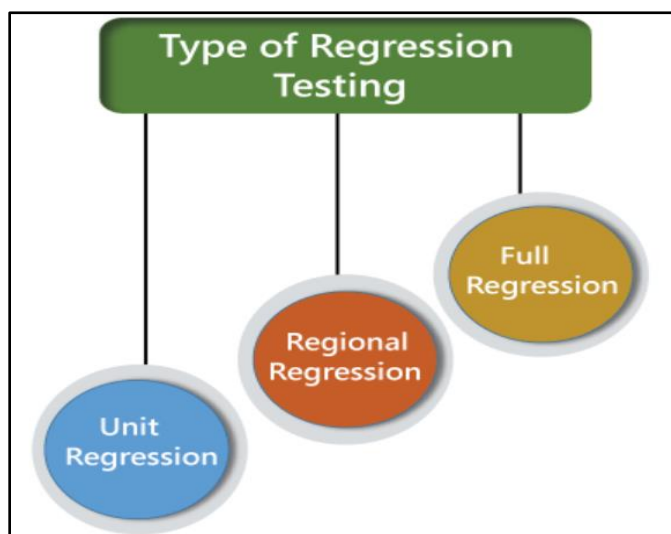
- Re-execution of same test cases in different releases/ Sprints/ Builds to make sure that changes (addition, modification, removal of features and Bug fixes) are not introducing defects in unchanged features is called as Regression Testing.

(OR)

- Testing the unchanged features to make sure that it is not broken because of changes is called as Regression Testing and here changes can be addition, modification, removal of features and Bug fixes.



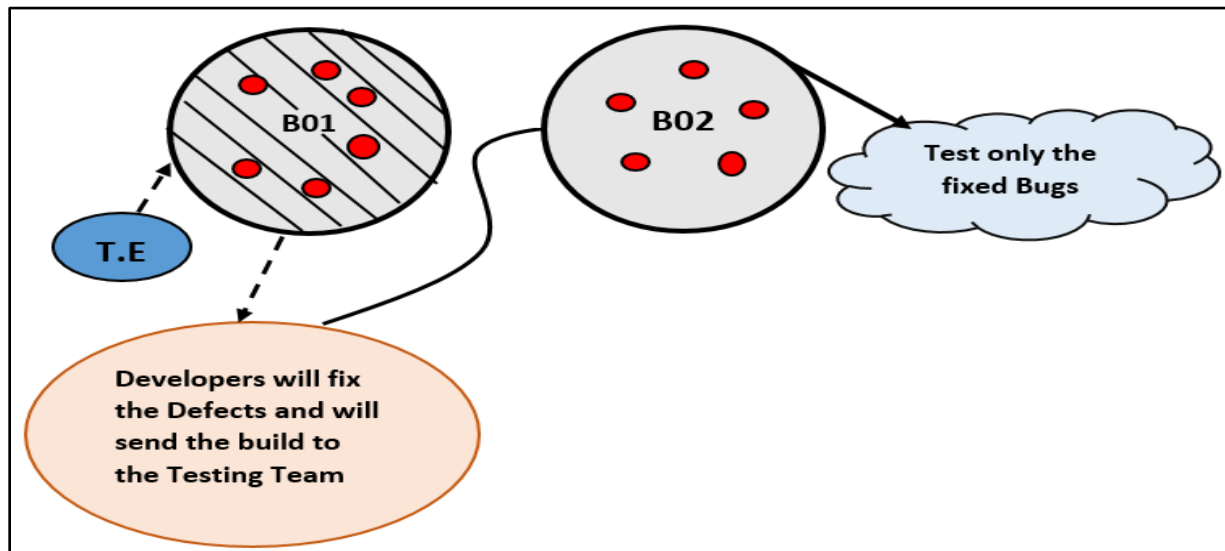
Types of Regression Testing:



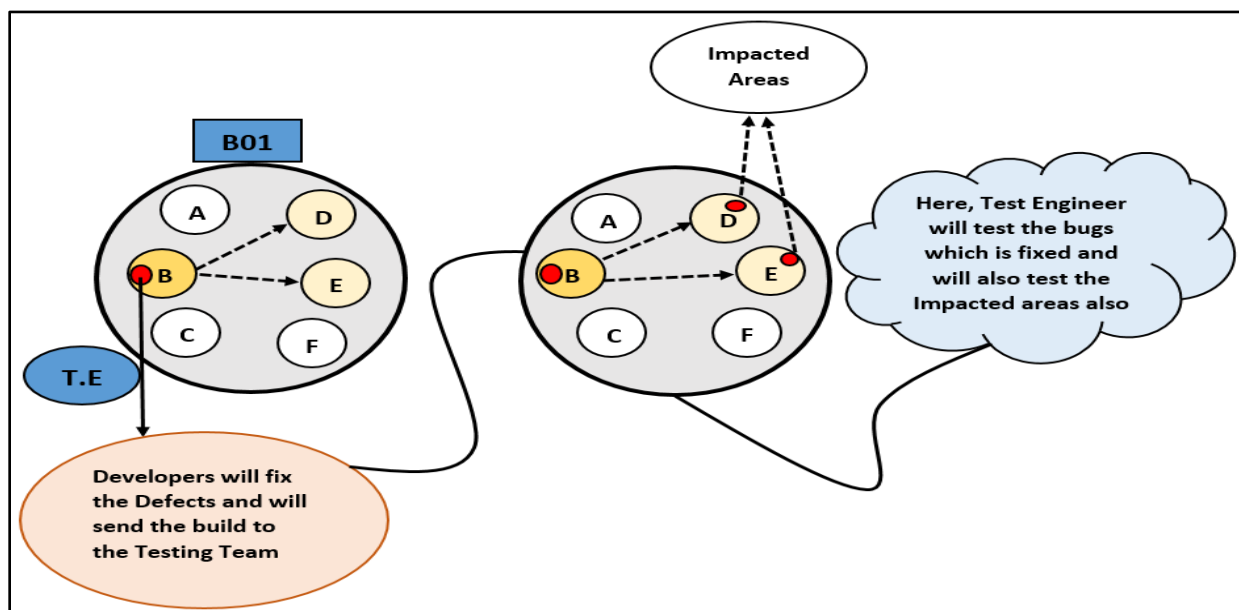
- Unit Regression Testing.
- Regional Regression Testing.

3. Full Regression Testing.

- ❖ **Unit Regression Testing:** Testing only the bug which is fixed or the changes made is called as Unit Regression Testing.



- ❖ **Regional Regression Testing:** Testing only the changes and all the impact regions is called as Regional Regression Testing.



- ❖ **Full Regression Testing:** Testing the changes and all the remaining features is called as Full Regression Testing.

How will you identify areas or how you will do Impact analysis?

- Based on the product knowledge. The Test Engineer will be knowing how each and every module works and also I will be knowing how each and modules are related. Based on that knowledge I will be able to identify the Impact areas.
- By preparing **Impact Analysis Matrix** we identify areas wherein we list the changes and also list all the features and also we map it.
- By conducting **Impact Analysis Meeting** here entire testing team meets and discuss about the list of changes, bug fixes and also associated impact region.

Impact Analysis Matrix:

Serial Number	Changes Made	Login	Inbox	Sent Item	Logout
1	Attachment	X	Y	Y	X
2	Password	Y	X	X	Y

Why or When we do Regression Testing?

- Whenever too many changes are done in the product.
- Whenever changes made in the Root of the product or in the core feature.
- At for last few cycles or release or the sprint we should do regression full regression testing.

Advantages of Regression Testing:

- By not testing all the features we save lot of time.
- This reduces test cycle duration.

Drawbacks of Regression Testing:

- Chances are there we might miss to identify impact areas and because of that we might miss a bug.

Difference between Retesting and Regression Testing:

RETESTING	REGRESSION TESTING
Checking whether the bug is really fixed or Testing only the changes made.	Doing the changes or fixing bugs that might have impact on the other features and testing the unchanged features to make sure that it is not affected by the broken modules.

19) What is Defect and Defect Tracking Process?

- Deviation from the requirement specification is called as Defect.
(OR)
- Any feature which is not working according to the requirement specification is called as Defect.
- **BUG:** Informal name given to the defects.
- **Error:** Error is the mistake done by the programmers in the program, due to which we are not able to compile or run the program. There are 2-types of error:
 - * **Compile Time Error:** We get compile time error because of Syntax Mistakes.
 - * **Run Time Error:** We get run time error because of Logical Mistakes.
- **Failure:** Defect in the software leads to failure. One defect might lead to one failure or multiple failure.

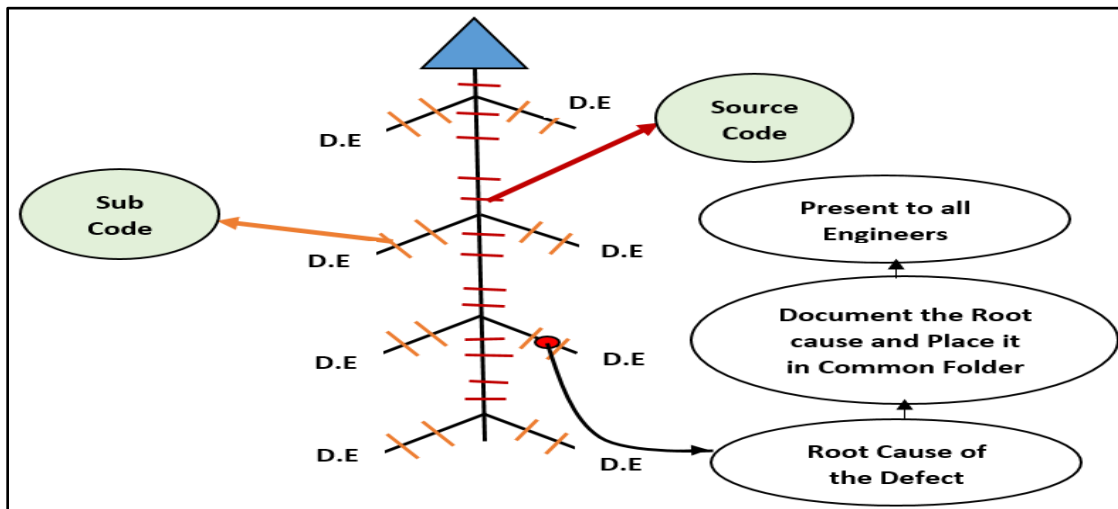
Why we Defects?

- **Because of Wrong Implementation:** Developers would have written wrong code.
- **Because of Missing Implementation:** Developers would have not written the code itself.
- **Extra Implementation:** Developers would have added extra features or modules.

Defect Tracking Process?

❖ Root Cause Analysis Meeting:

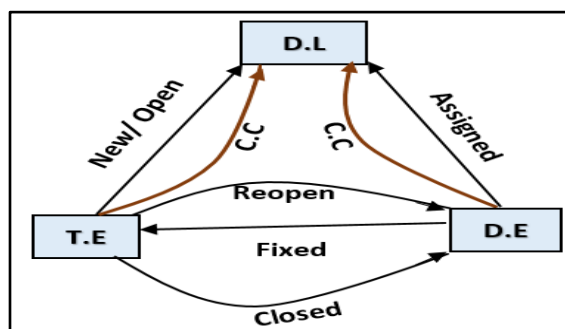
- Test Engineer while testing the software, if finds any defects the Test Engineer will communicate the defects to the Developers.
- Developers will sit together and try to find the “**Root cause of the Defect**” and will document the root cause and keep it in a common folder where everybody in the team can access the program and can present the document to the entire team.
- This Process is called as “**FISH BONE TECHNIQUE**” (or) “**ISHIKAVA METHOD**” (or) “**Root Cause Analysis**” (RCA) meeting.



Test Efficiency: Test Efficiency is given as,

$$\frac{\text{Total Number of Defects found by Test Engineer}}{\text{Total Number of Defects found by Test Engineer} + \text{Total Number of Defects found by Customers} + \text{Number of defects found by Users}} \times 100$$

Defect Tracking Process:



1. Test Engineer (T.E):

- T.E will find the defect.
- T.E will login into defect tracking tool and will prepare defect report.
- T.E will put status as **New/ Open**.
- T.E will send this report to Development Lead.

2. Development Lead (D.L):

- D.L will read the report and understand the problem.
- D.L will identify the developer who has done the mistake and will assign the defect.
- D.L will change status to **Assigned**.
- D.L will send report to Development Engineer.

3. Development Engineer (D.E):

- D.E will read the report and understand the problem.
- D.E will go to source code and fix the defect.
- D.E will change the status as **Fixed**.
- D.E will send that report to Test Engineer and also keep's CC for Development Lead.

4. Test Engineer (T.E):

- T.E will read the report and understand the defect.
- T.E will Retest the bug if it is fixed he will change status as **Closed**.
- If the defect is not fixed he will change the status as **Reopen**.

Why Test Engineer should not wait for the permission of test lead to send bug report to development lead?

- There will be delay in communicating Defect report (Defects).
- As a T.E, I will be having knowledge in depth about my feature so it is better take a decision and send report to Development Engineer and make sure that you are keeping CC to the Test Lead.

Why Test Engineer should keep CC for test lead?

- Test Lead is a one who attends all types of meeting like customer management and development team he should be aware of all thing that are there in product.
- To get visibility that Test Engineer is working.

As soon as you get defect immediately you should send that to Developer. Why?

- Test Engineer might forget the defect.
- Some other Test Engineer might send your defect.
- If the defect is sent early developers will have sufficient time to fix the defect.

20) What is SEVERITY and PRIORITY?

❖ **SEVERITY:** It is decided based on the impact of defect on customer business.

There are different levels of severity:

- * **Blocker or Showstopper Defect:** Assume that there is a defect in the software and I am 100% sure that this defect will affect the customer business flow and also it is blocking TE to test the application. Such type of defect is called as Blocker or Showstopper defect.
- * **Critical Defect:** Assume that there is a defect in the software and I am 100% sure that this defect will affect the customer business flow, but Test Engineer is not blocked to test the feature. Such defect is called as critical defect.

- * **Major Defect:** Assume that there is a defect in the software (application) and I am not sure that how this defect will affect the customer business work flow. This kind of defect is called as Major defect.
- * **Minor or Trivial Defect:** Assume that there is a defect in the application and I am 100% sure that this defect will not affect the customer business flow. This kind of defect is called as Minor defect.

❖ **PRIORITY:** Importance given to fix the defect or how soon defect must be fixed by the developer.

There are different levels of priority:

1. **High or P1:** If the defect is having High priority (P1), then developer should immediately fix the defect.
2. **Medium or P2:** If the defect is having priority as Medium (P2), then developers should fix the defect within some test cycles (or) within some builds (or) within a release.
3. **Low or P3:** If the defect is having priority as Low (P3), then developers can fix the defect in upcoming release or within 2 to 3 releases.

There are 4 combinations:

- a. High Severity and High Priority
- b. High Severity and Low Priority
- c. Low Severity and Low Priority
- d. Low Severity and High Priority

HS and LP:

1. In Face Book help page is navigating to blank page.
2. If WhatsApp is not installed for the 50th time.
3. In WhatsApp invite share is not working for one of the 3rd party application.

LS and HP:

1. Spelling mistake in the welcome page (Facebook).
2. Login becomes Loving.
3. Alignment issue in the login page.

Who will give severity and priority?

- Test Engineer will give the severity and priority.
- One defect will have only one severity and one priority.

✓ **Note:** Developers will always fix the defects which is having **Highest Priority** rather than **Highest Severity**.

Assignment: Identify Severity and Priority?

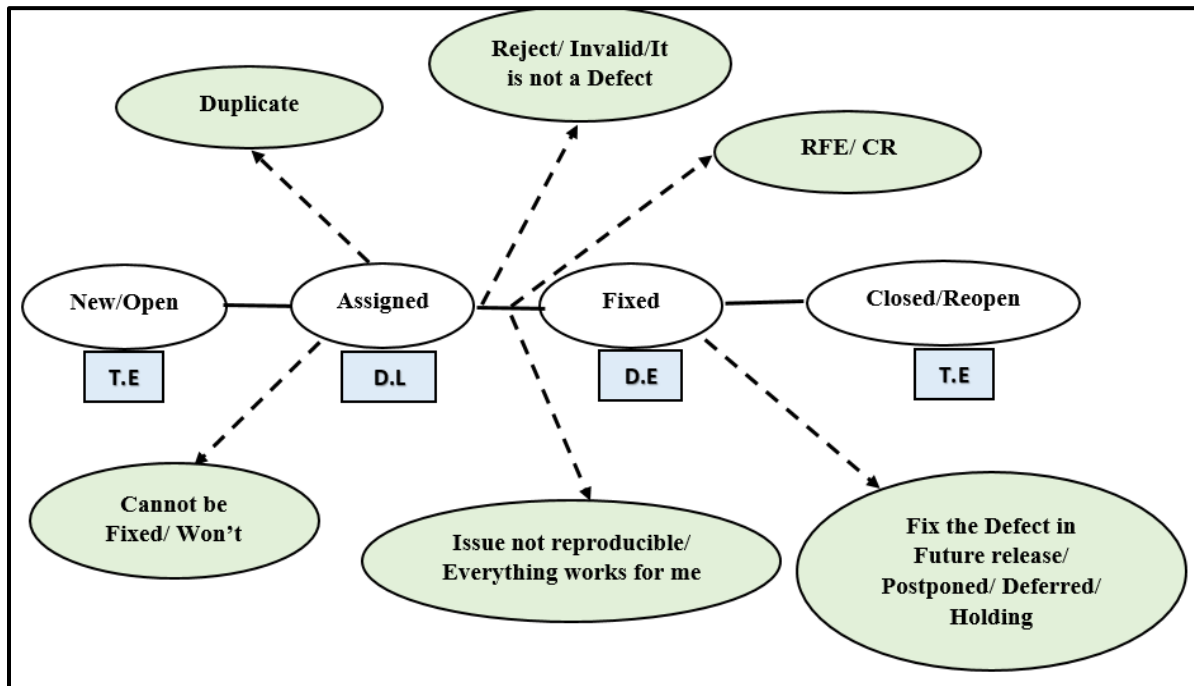
- 1) In WhatsApp user is not able to do call.
- 2) In WhatsApp user is not able to do Video call.
- 3) In ATM machine, User is able to give wrong pin and is able to withdraw the amount from the ATM.
- 4) In OLA, User is able to book a ride and user finishes the ride by providing wrong OTP.
- 5) In FB, Remember Password feature is not working.

- 6) In Amazon, In Settings, there are spelling mistakes.
- 7) **Give Examples of your own: 5-Blocker, 5-Critical, 5-Major, 5-Minor.**

✓ **Note:** **Release:** Starting from gathering the Requirement, Developing the product (software or application) and Testing it for many cycles until we release or deploy product to the Production is called as One Release. If the company follows AGILE Model, then the above process is called as “**SPRINT**”.

✓ **Note:** **Test Cycle:** It is the effort spent or time spent by the Test Engineers to test the software.

21) What is Defect Life Cycle/ Bug Life Cycle?



1) Test Engineer (T.E):

- T.E will find the defect.
- T.E will login into defect tracking tool and will prepare defect report.
- T.E will put status as **New/ Open**.
- T.E will send this report to Development Lead.

2) Development Lead (D.L):

- D.L will read the report and understand the problem.
- D.L will identify the developer who has done the mistake and will assign the defect.
- D.L will change status to **Assigned**.
- D.L will send report to Development Engineer.

3) Development Engineer (D.E):

- D.E will read the report and understand the problem.
- D.E will go to source code and fix the defect.
- D.E will change the status as **Fixed**.
- D.E will send that report to Test Engineer and also keep's CC for Development Lead.

4) Test Engineer (T.E):

- T.E will read the report and understand the defect.
- T.E will Retest the bug if it is fixed he will change status as **Closed**.
- If the defect is not fixed he will change the status as **Reopen**.

Defect life cycle consists of below mentioned status.

- 5) **New:** Test Engineer when he finds defect for the first time status will be “New”.
- 6) **Open:** When Developer start to work on the defect then the status will be in open.
- 7) **Reject/ Invalid/ It is not a Defect:** Test Engineer will assume feature itself as defect and he will send that defect to developer. Developers say that it is not a defect it is feature in such case they will change the status to Reject/ Invalid/ Not a defect.

Why do we get Reject status?

- Because of Misunderstanding the requirement.
 - Because of extra feature.
- ✓ **Note:** Whenever Test Engineer calls extra feature as defect chances are there developers might Reject it. In such case Reopen the defect and ask to update the requirement.
- When the build or software is wrongly configured or wrongly installed, If Test Engineer install the build wrongly and find the defect in the software and communicate that to developer, developer say that it is not a defect because code is perfect and TE is not properly installed.
 - Because of referring old requirement.
- 8) **Duplicate:** If Test Engineer finds defect and communicate defect to the developer and if same defect is logged/ tracked by other Test Engineer then Developers will tell that this is duplicate of old defect.

Why we get Duplicate status?

- Because of testing common feature.
- Old TE would have found lot of defects in that there are still some pending bugs which has to be fixed. If new Test Engineer join to same project and send's same bugs in such case developer say that it is duplicate.
 - * **Case 1:** Test Engineer will find the defect in the software if already same defect is present with the status as “New” in this case Test Engineer should never track the defect.
 - * **Case 2:** Test Engineer finds defect in the software if already same defect is present with the status as “Fixed” then Test Engineer Should Reopen the defect.
 - * **Case 3:** Test Engineer finds defect in the software if already same defect is present with the status as “Closed” in this case Test Engineer should track as New defect.

How to avoid Duplicate status?

- Test Engineer when he finds defects communicate defect to the developers and will keep CC for Test Lead and also he should keep CC for all Test Engineers who are working in the same project.
- Test Engineer when he finds defect, before preparing a report Test Engineer should login into the defect tracking tool and should search for duplicate defects in Defect Tracking Tool by entering certain keywords in “Advance Search”.
- Test Engineer when he finds defect before communicating that defect to Development Engineers Test Engineer should cross check with other Test Engineers, Test Lead and also developers so that you can avoid duplicate.

Why developer will say it as duplicate?

- To reduce defect count.
- To reduce duplicate effort.

9) **Cannot Be Fixed/ Won't Fix:** Developers are accepting it as a defect but they are not in the position to fix the defect. In such case developers will change status as cannot be fixed.

Why we get Cannot Be Fixed/ Won't Fix Form Developers?

- If there is a minor defect present in the root of the software and if it not is affecting customer business, then developer say it as Cannot be fixed.
 - Because of the technology is not supporting: It means the programming language which is used to develop the software has got no capacity to fix the problem.
 - If the cost of fixing the bug is more than the cost of the Bug, that is cost of the bug means it the loss in the business because of having the bug in the software.
- 10) **Postponed/ Deferred:** Developers are accepting that it is the defect they want to fix it little later, in this case developer will change status as Postponed.

Why?

- If Test Engineer finds minor defect at the end of the release and if developer is not having sufficient time to fix that defect, then developer will say it as Postponed.
- If Test Engineer find defect in the feature which is not necessary for the customer in the current release, then developer will say it as Postponed.
- If Test Engineer finds a defect and send it to developer and developer will say that customer is expecting changes in the same feature so this defect is postponed until we get clarity from the customer.
- If the Test Engineer finds the defects which is exposed to the internal users, then developer will say that he will fix it in Future release.

- 11) **Issue is not Reproducible/ Everything works for me:** Test Engineer is able to see the defect but developers are not able to see the same defect in such case developers say that Issue is not reproducible.

Why?

- Because of platform mismatch:
 - Because of OS mismatch
 - Because of browser mismatch
 - Because of browser version mismatch
 - Because of browser settings mismatch
 - Because of improper defect report.
 - Because of incorrect data.
 - Because of inconsistent defect.
- ✓ **Note:** **Inconsistent Defect:** If the feature is not working for time and same feature is working for some other time, then these kind of defects are called as Inconsistent defects. If Test Engineer finds these kind of defects, then tracking the defect, Test Engineer should put a note saying it as Inconsistent Defects.

- 12) **RFE (Request for Enhancement)/ CR (Change Request):** While testing software if test engineer finds any defect which is not a part of the requirement. Then developer will say that defect as RFE.

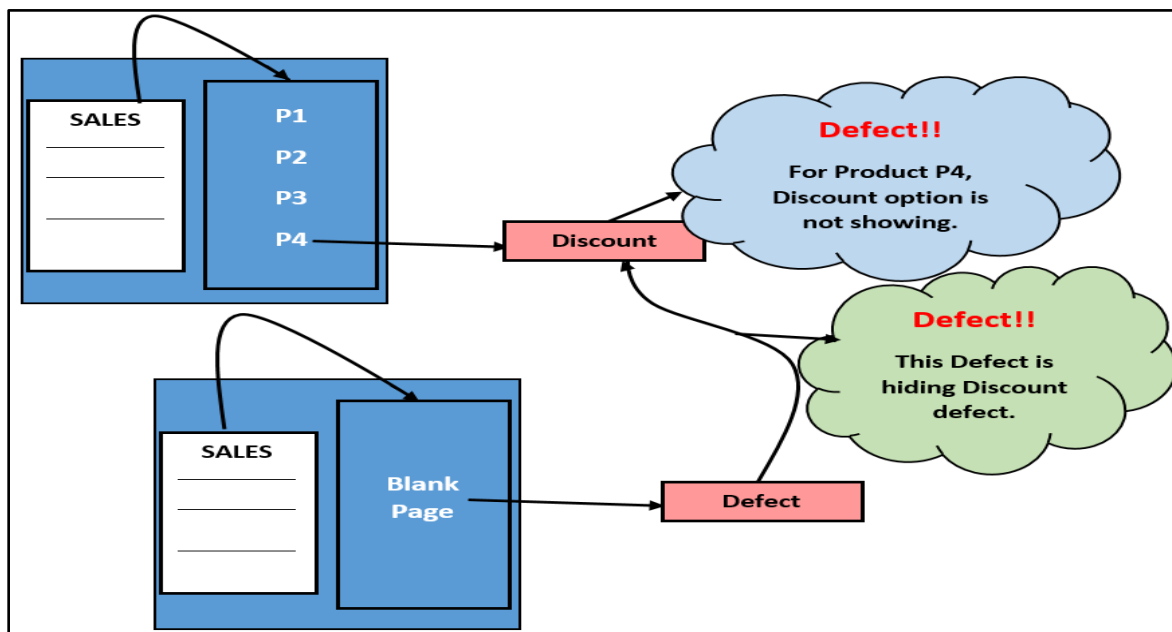
Who can give RFE?

It can be given by customer/ Test Engineer/Development Engineer/ Test Lead.

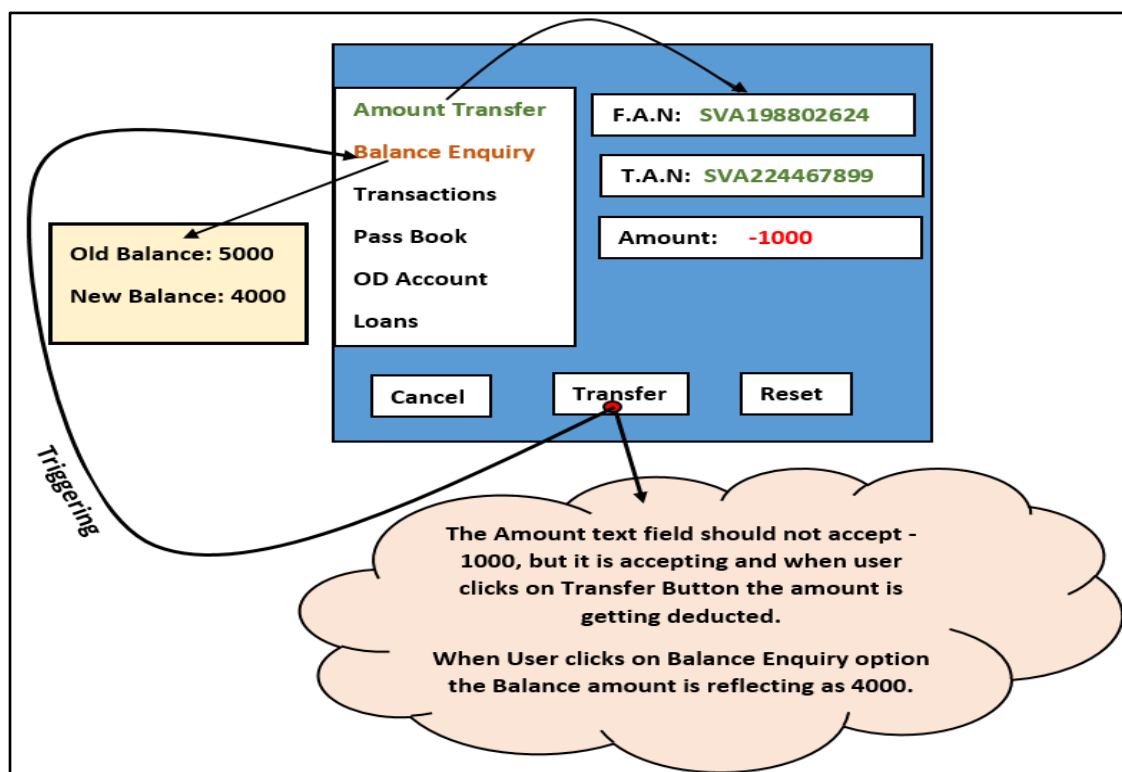
22) What is?

- 1) **Defect Triage:** When time is very less and when there are too many pending bugs to be fixed, then we categorized the defects into different status.
- All the **Business Critical** defects are moved to **Assigned Status**.
 - There are some defects which need not be fixed on urgent basis, they move such defects to **Postponed Status**.
 - There are defects which need not to be fixed (Customer is ready to accept the product with the defects), they move such defects to **Cannot be Fixed Status**.
- 2) **Defect Seeding:** Intentionally introducing defects in the software to check the effectiveness of the Test Engineer is called as Defect seeding.

3) **Defect Masking:** Any defect, hiding another defect called as Defect Masking.



4) **Defect Cascading:** Any defect which is triggering another defect is called as Defect Cascading.



5) **Defect Clustering:** When the small number of modules contains most of the defects or test failures is called as Defect Clustering. According to **PARETO** principle, approximately 80% of the defects will be present in 20% of the modules.

- 6) **Pesticide Paradox:** Executing the same Test cases again and again which will not catch any new defects, to overcome this drawback we should review and modifying the Test cases and reviewing it again in order to find hidden defects is called as Pesticide Paradox. When 20% of the modules having maximum number defects is called as **“Hot Spots”**.
- 7) **Latent Defect:** It is a defect which is carried forward in every sprint but, which has to be fixed in earlier stage itself.
- 8) **Defect Leakage:** The Defects that have been missed by Test Engineers while testing the software.
- 9) **Re-Spin:** Getting one more build within one test cycle is called as re-spin.

23) Different types of Defect Tracking tools?

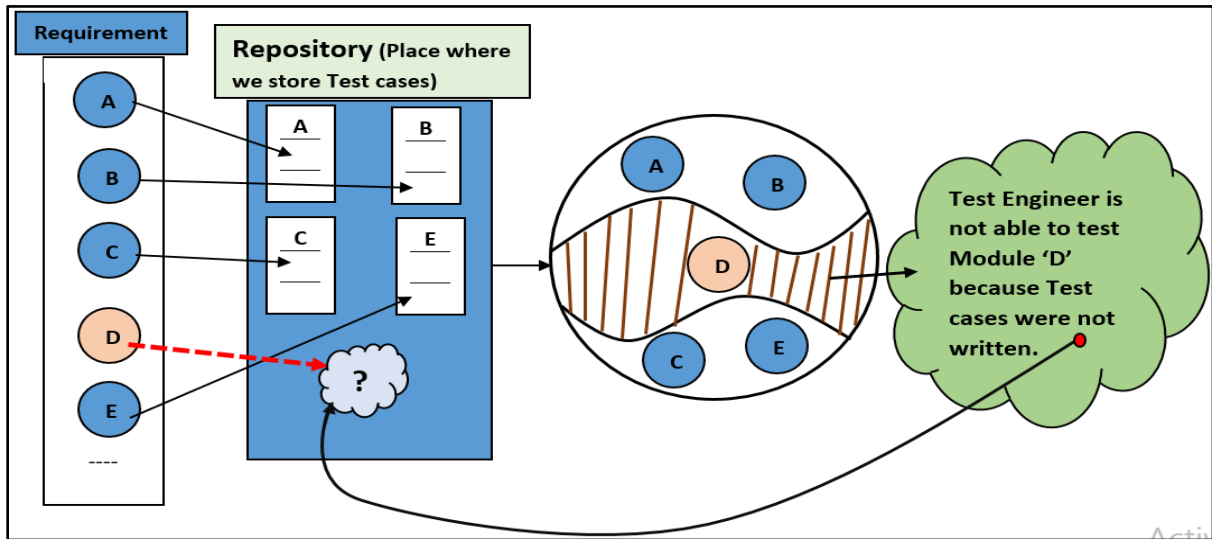
- Defect Tracking Tool is a software, which is mainly used to track (or) store the defects in a centralized place and communicate the defect to developers in an organized way.
- Different Types of Defect Tracking Tools:
 - * **BUGZILLA**
 - * **QC/ALM (PMT)** (Quality Controller/ Application Lifecycle Management)
 - * **JIRA (PMT)**
 - * **BUGZINI**
 - * **BUGPRO**
 - * **BUGNET**
 - * **MANTIS**
 - * **CROCPLUS**

Defect Template:

Defect ID	Module Name	Build Number	Test Case_ID	Defect Description	Test_Data	Steps to Reproduce	Severity	Priority	Status	Screen Shots/ Screen Recordings	Platform
D_LN_001	Login Page	B01	TC_LN_003	User is not able to login to the application.	UN: Kevin Pwd:HJ&*88	1. Open the "Browser". 2. Enter the "URL". 3. Click on "Login" link. 4. Click on "User Name" text field. 5. Provide valid User Name into the User Name text field. 6. Click on "Password" text field. 7. Provide valid Password into the Passowrd text field. 8. Click on "Login" button.	Critical	P1	New/ Open		Windows 10

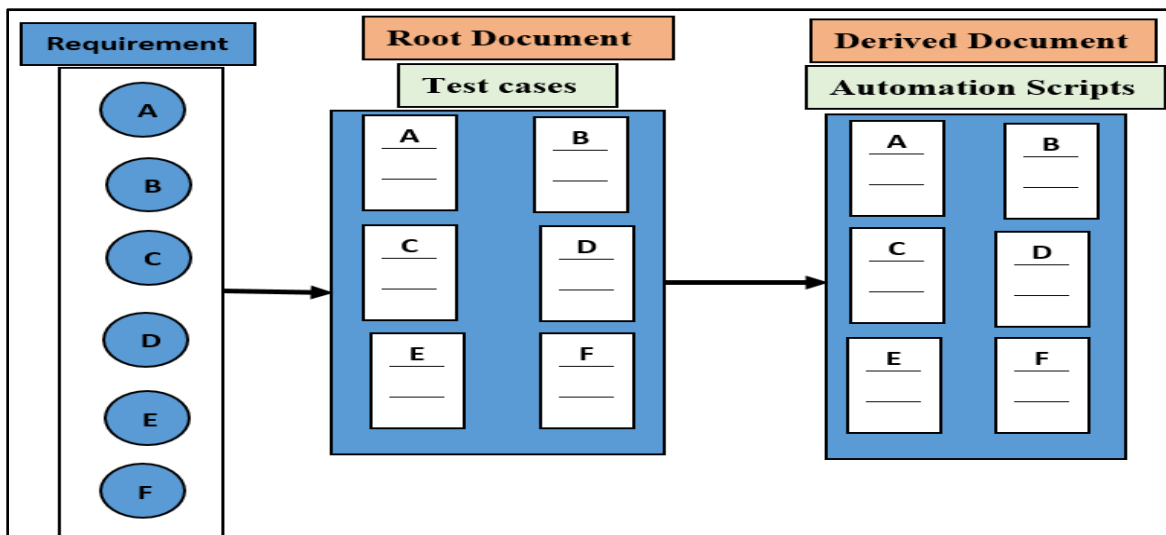
24) What is Traceability Matrix?

- Once test cases are ready the biggest question is what is the guarantee that you have written test cases for all the modules to ensure that we prepare traceability matrix.
- Traceability matrix is a document which ensures that every requirement has got at least one test case.



Types of Traceability Matrix:

- There are Three types of Traceability Matrix.
 - * **Forward Traceability Matrix.**
 - * **Backward Traceability Matrix.**
 - * **Bi-directional Traceability Matrix.**



- 1) **Forward Traceability Matrix (F.T.M):** Mapping from Root Document to Derived Document is called as Forward Traceability Matrix.
- 2) **Backward Traceability Matrix (B.T.M):** Mapping from Derived Document to Root Document is called as Backward Traceability Matrix.
- 3) **Bi- Directional Traceability Matrix:** Following both FTM and BTM is called as Bi-directional Traceability matrix.

Advantages of Traceability Matrix:

- It ensures that every requirement has got at least one test case which indirectly assures that you are testing every feature at least one.
- It gives traceability from high level requirement to automation scripts.

25) What is Test Plan?

Test plan is a document which drives the future testing activities. It has got different section like:

- 1) **Objectives:** This section covers aim to prepare test plan.
- 2) **Effort Estimation:** This section covers the estimation of how many engineers are required to complete the project, how long does it takes to complete and also what is the cost required to complete the testing.
- 3) **Scope:** This section covers what are all the features to be tested and what are all features not to be tested.
- 4) **Approach:** This section covers how we go about to test the product in future.
- 5) **Assumption:** This section covers the assumption that we have made while planning. We are planning means, we are assuming that we will do lot of task and based on that we will promise customer.
- 6) **Risk:** If any assumption fails that becomes risk, that is because of few reasons if we are not able to do certain task, then this becomes Risk.
- 7) **Mitigation plan/Backup plan/contingency plan:** In order to face the Risk, we have a plane which is called as Mitigation Plane.
- 8) **Test Methodologies:** This section covers what are all the types of testing that we are going to conduct in future.
- 9) **Schedule:** This section covers when exactly which activity should start and when exactly which activity should end.
 - * Smoke Testing
 - * Functionality Testing
 - * Integration Testing
 - * System Testing
 - * Usability Testing
 - * Regression Testing
 - * Adhoc Testing
 - * Performance Testing
 - * Compatibility Testing
 - * Globalization Testing

10) Test Environment: This section covers how we go about setting up the test environment in future.

11) Defect tracking: This section covers how to track a defect and also it covers procedure, severity, priority levels and also the defect tracking tool which we are planning to use.

12) Test Automation: This section covers what are features to be automated and what are all the features not to be automated and the complete automation strategy.

13) Deliverables: This section covers what are all document or outcome that should be given by testing team at the end of test life cycle.

✓ **Release Note:** Along with the product we release one note to the customer that is called as release note. It consists of:

- i. List of all known defects or open defects.
- ii. List of the platform in which product is tested.
- iii. List of the platform in which product is not tested.
- iv. List of defects found in previous release and fixed in the current release.
- v. Installation steps and also version number.

14) Entry and Exit Criteria:

- Entry Criteria: It is a list of criteria that has to be met to start the activity.
- Exit Criteria: It is the list of criteria that has to be met to say that activity is over.

15) Test stop Criteria:

This section covers when to stop testing. We stop testing in two different cases that is:

✓ **Product quality should be very good:**

- All the features requested by the customer should be ready.
- All end to end business scenarios should work fine.
- There should be zero blockers and zero critical bugs.
- If there are any bugs pending that should be within the acceptable limit set by the customer.
- You should have tested in the environment similar to production.

✓ **Product quality should be very bad:**

- If there are too many blockers and critical.
- If many end to end scenarios are not working.
- If it is crossing budgets.
- If it is crossing the schedule.

16) Roles and Responsibilities: This section covers what each engineer should do in different stages of test life cycles.

17) Templates: This section covers format for all the documents that you are planning to prepare in the entire test life cycle.

26) What is Test Case?

- Test Case is a document which covers all possible scenarios on a specific requirement.

Why do we write Test Case?

- **To have consistency in the test execution:** If you document all the scenario you can make sure that you are executing all the scenario in all the test cycles and in all the sprints.
- **To have better test coverage:** When the requirement comes developers are busy in developing product at the same time testing team should write test cases and cover all all possible scenarios so that when build comes for testing we don't miss any scenarios.
- To depend on process rather than person.
- To avoid training every new engineer on the product or on the requirement.
- Test cases is the only document which acts like proof to your customer, developing team and manager that you have covered all possible scenario.
- **Test case acts like base document for writing the automation script:** If you refer the test case and then automate the script you can ensure that same kind of coverage is there even in automation script.
- If you don't document the test cases, you will forget the scenarios.
- If you document all scenario, then test execution happens in very organised way.
- If you have written or documented test cases, the test execution time will be less.

When do we write Test Case?

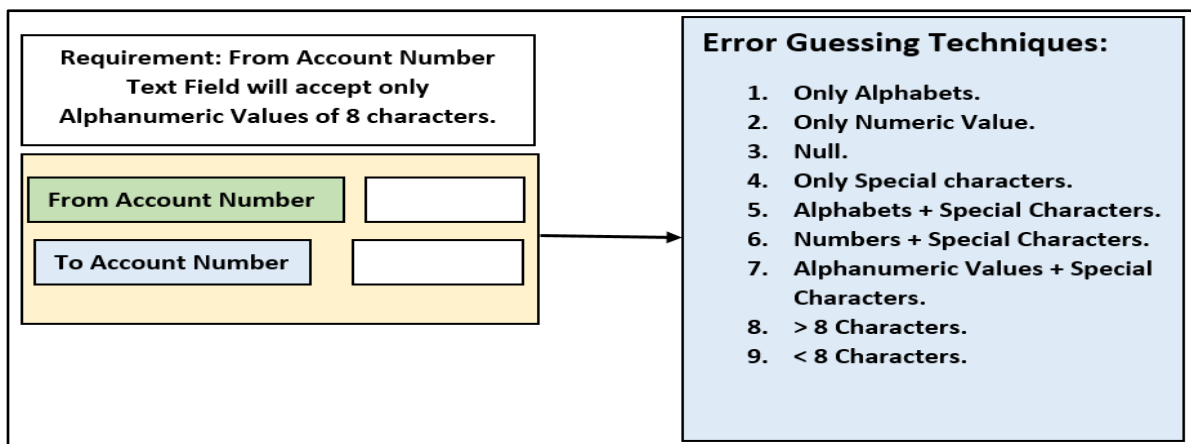
- When developers are building the product, testing team will be writing test cases.
- When developers are adding the features, test engineer will be adding test cases.
- When developers are modifying features, test engineer will be busy in modifying the test cases.

Test Case Template:

Project Name										
Module Name										
Author										
Date of Creation										
TestCase_ID	Module Name	Test Case Type	Pre-Requisites	Scenario Description	Test Steps	Test data	Expected Result	Actual Result	Status	Comments
TC_Registration_001	Registration	Positive	URL should be in working condition	Validate that user is navigated to Register Page when user clicks on "Register" Button.	1. Open the Browser. 2. Enter URL. 3. Click on Register Button.	URL: https://www.booking.com	User should be navigated to Registration Page.			
TC_Registration_002	Registration	Positive	URL should be in working condition	Validate that user is able to click on "E-mail ID" Text field.	1. Open the Browser 2. Enter URL 3. Click on Register Button 4. Click on E-mail ID Text field.	URL: https://account.booking.com/registration	User Should be able to click on E-mail ID textfield.			
TC_Registration_003	Registration	Positive	URL should be in working condition Email ID should be existing in the Data base	Validate that user is able to enter valid E-mail ID into "E-mail ID" Text field.	1. Open the Browser 2. Enter URL 3. Click on Register Button 4. Click on E-mail ID Text field. 5. Enter valid E-mail ID.	URL: https://account.booking.com/registration E-mail: abcd455@gmail.com	User should be able to enter valid Email-Id into the Text field.			
TC_Registration_004	Registration	Negative	URL should be in working condition Email ID should be existing in the Data base	Validate that user is able to enter invalid "E-mail ID" Text field.	1. Open the Browser 2. Enter URL 3. Click on Register Button 4. Click on E-mail ID Text field. 5. Enter invalid E-mail ID.	URL: https://account.booking.com/registration E-mail: 123456 abcd@3344 @&*%*(&## 4464frhs\$%\$%\$%\$	User should not be able to enter invalid data into the Textfield.			
Reviewer										
Date of Review										

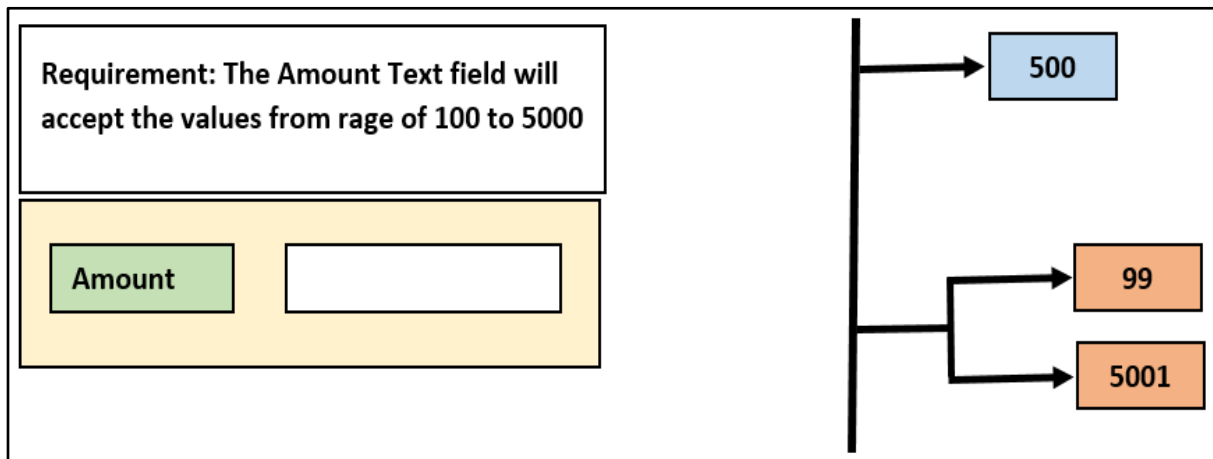
27) Test Case Design Techniques?

- 1) **Error Guessing Techniques:** Here we will guess all possible errors or defects based on that we derive scenarios, here we guess errors based on the requirement.



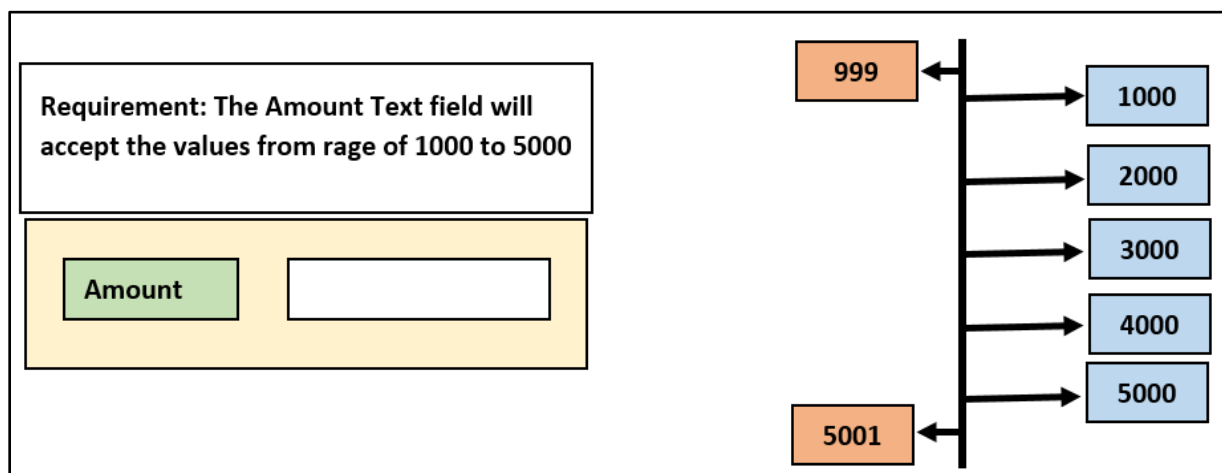
2) Equivalence Portioning.

- a) **Press Men:** If the input is ranged values then design the test case for one valid and two invalid value.

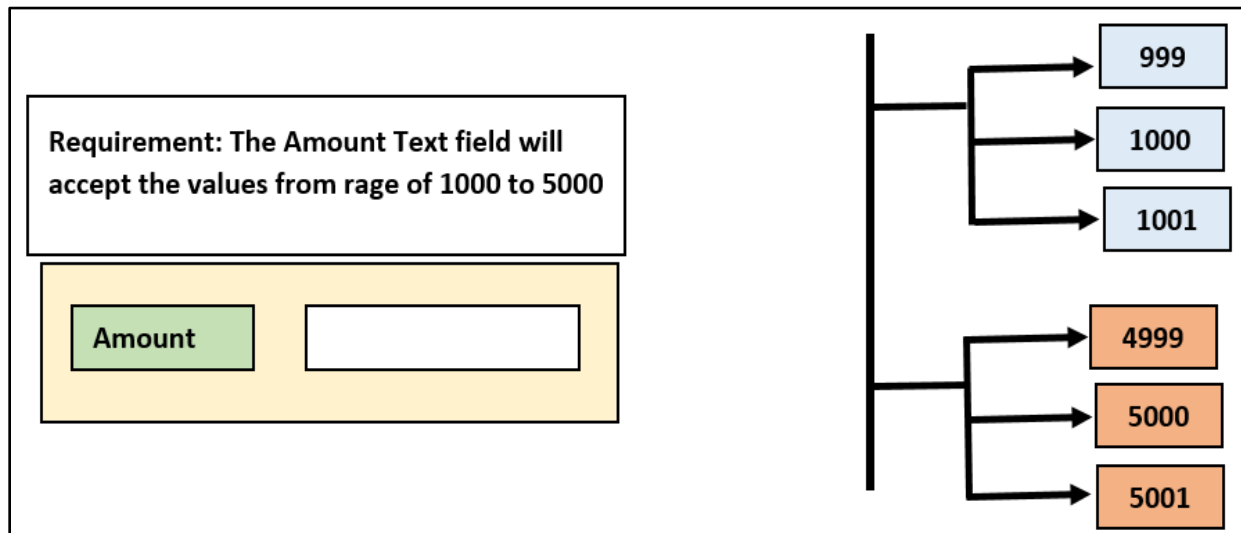


- If the Input is Boolean Values, then design the Test Case for both True and False Values.

- b) **Practice:** If the input is ranged values, Divide the range into equivalent parts and test for all the values but make sure that you are testing for at least two invalid values.



- 3) **Boundary Value Analysis:** If the Input is range of values between “A” to “B”, then design the test case for ‘A’, ‘A+1’, ‘A-1’ and ‘B’, ‘B+1’, ‘B-1’.




28) Test Case Review Techniques?

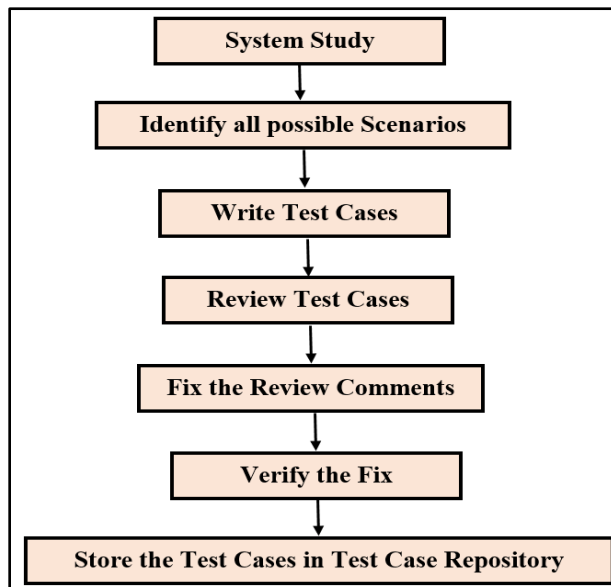
- I will go to the body of the test case and will look for: Missing scenario, Wrong scenario and Extra scenario.
- I will also check whether test case design techniques are applied or not.
- I will check whether scenarios are organised properly or not, so that it takes less time for the execution.
- I will check whether it is easy to understand so that the new engineer will be execute without asking any questions.
- I will look into the header and check whether all attributes covered or not and I will also check whether all attributes are having relevant content.

Review Ethics:

- * Always review the content not the Author.
- * While Reviewing, spend time only in finding the mistakes not the solution for it.
- * Even after the review if there are any mistakes, both the author and reviewer are responsible.

-  **Test Case Optimization:** When we apply Test Case Design Techniques, there will be many Scenarios and chances are there that many scenarios will be repeated, in order to avoid repeated scenarios, we go with Test Case Optimization.

29) Procedure to write Test cases (Procedure to write Test Case)?

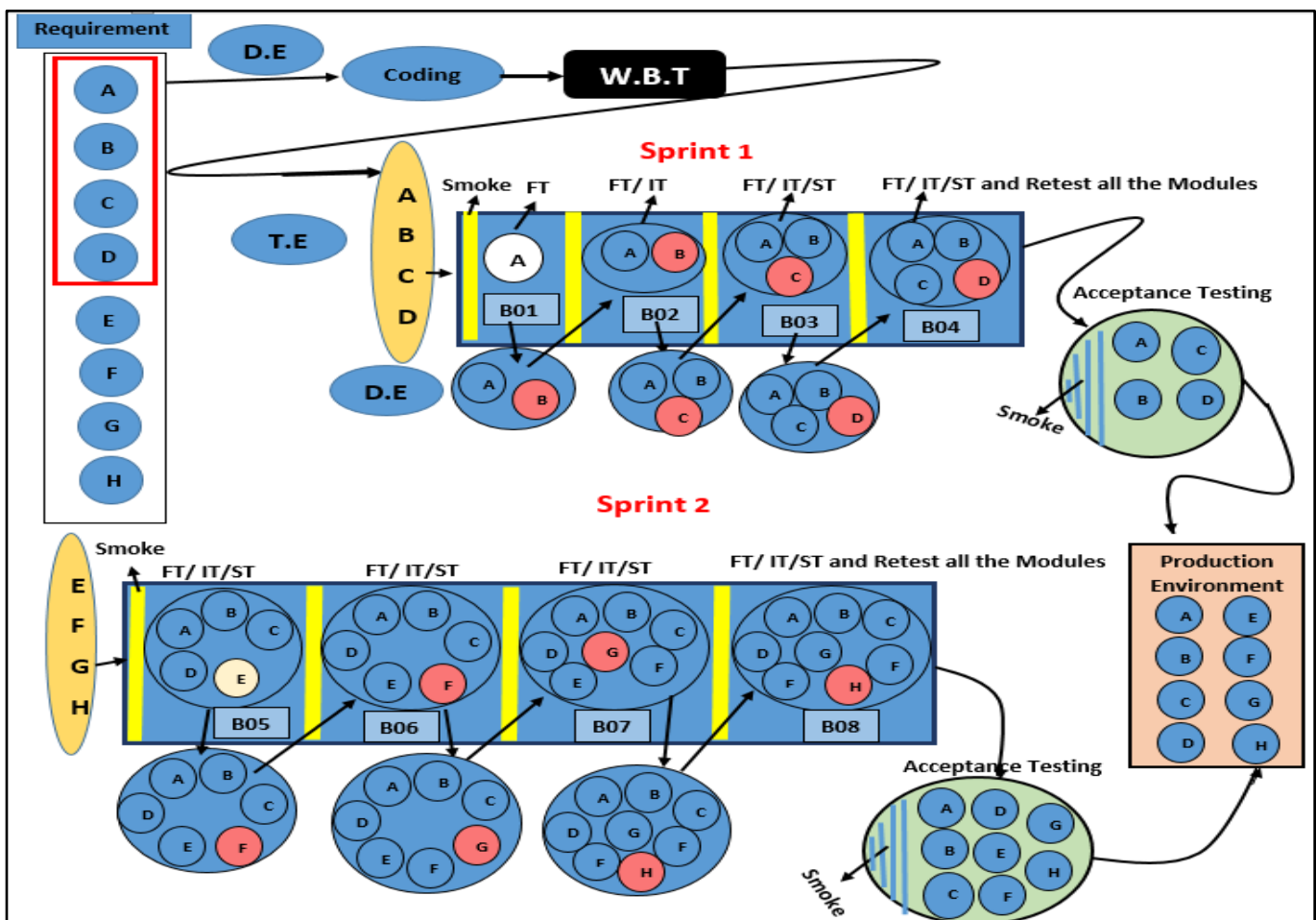


30) Agile Methodology?

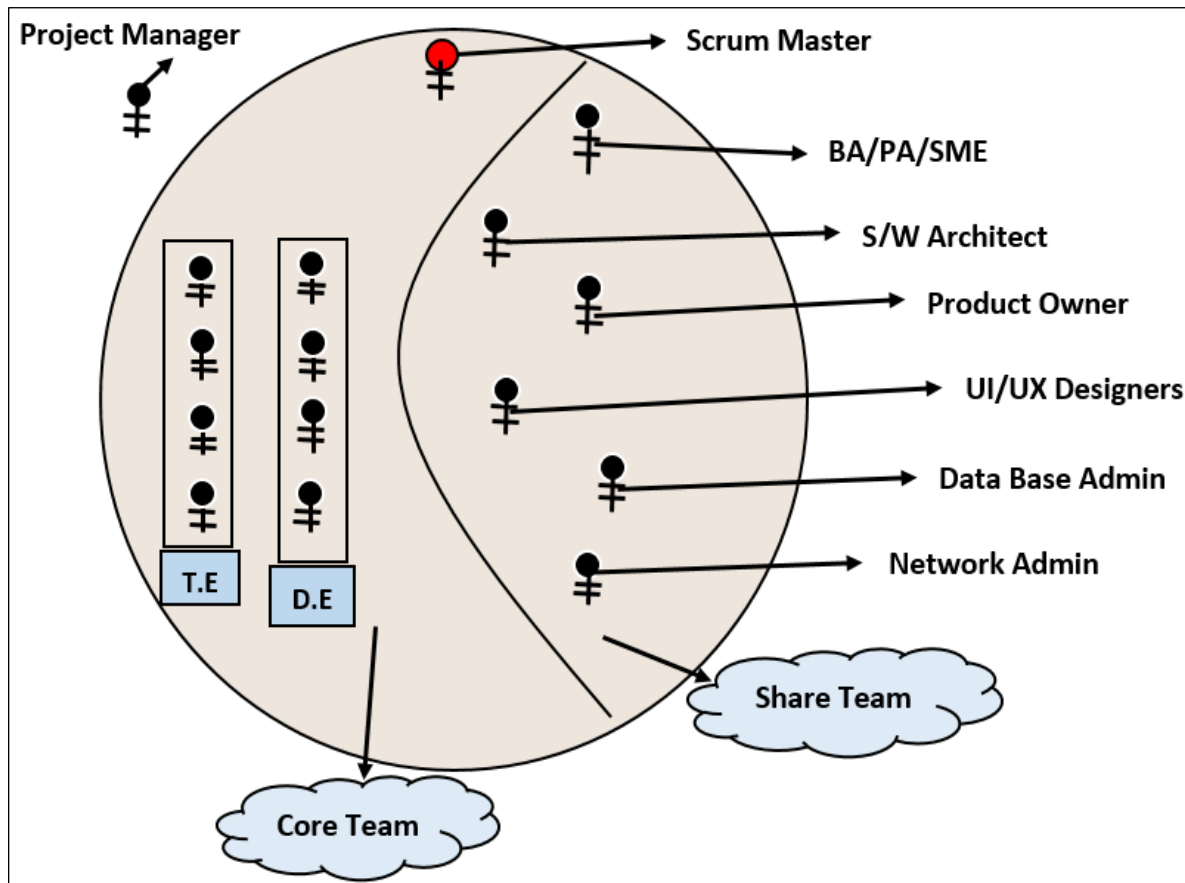
It is the model wherein we develop the software in an incremental and iterative manner. It is followed to overcome the drawbacks of the traditional model. Here they build large product in shorter cycles.

Sprint: Starting from collecting the requirement doing sprint planning, actually developing and testing the application for many cycles is called as One sprint.

Scrum: It is a process where in we used to build the software by following Agile.



1. Formation of Scrum Team:



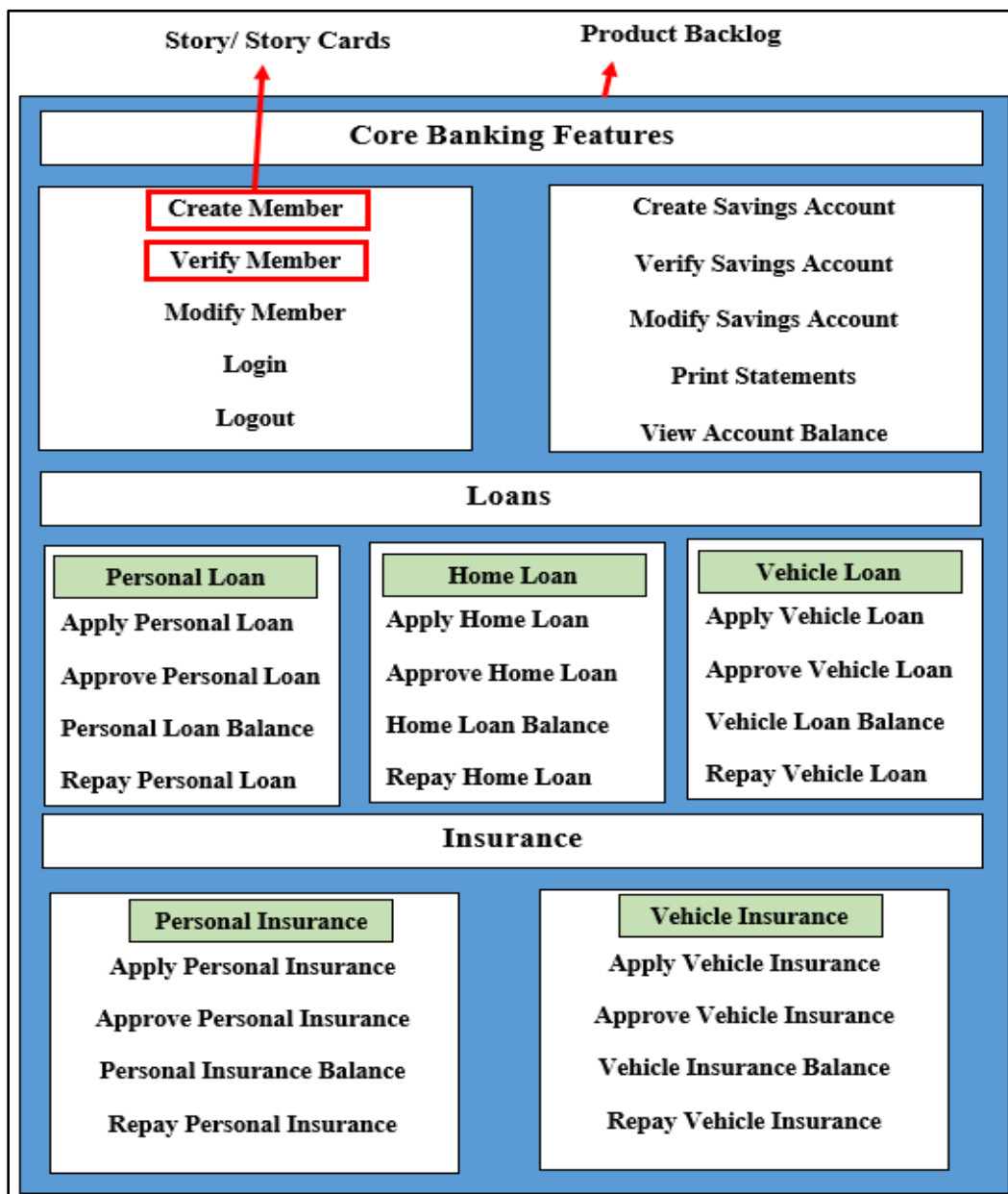
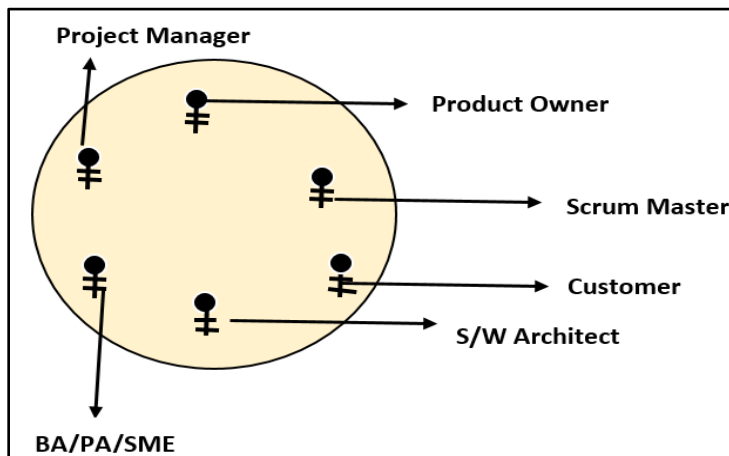
- **Scrum team:** Group of Engineers working towards completing the committed features or stories.
- Generally, scrum team will have 5 to 12 people.
- It includes Share team and Core team.
- **Core team:** Scrum Master, Test Engineer and Development Engineer.

Share team:

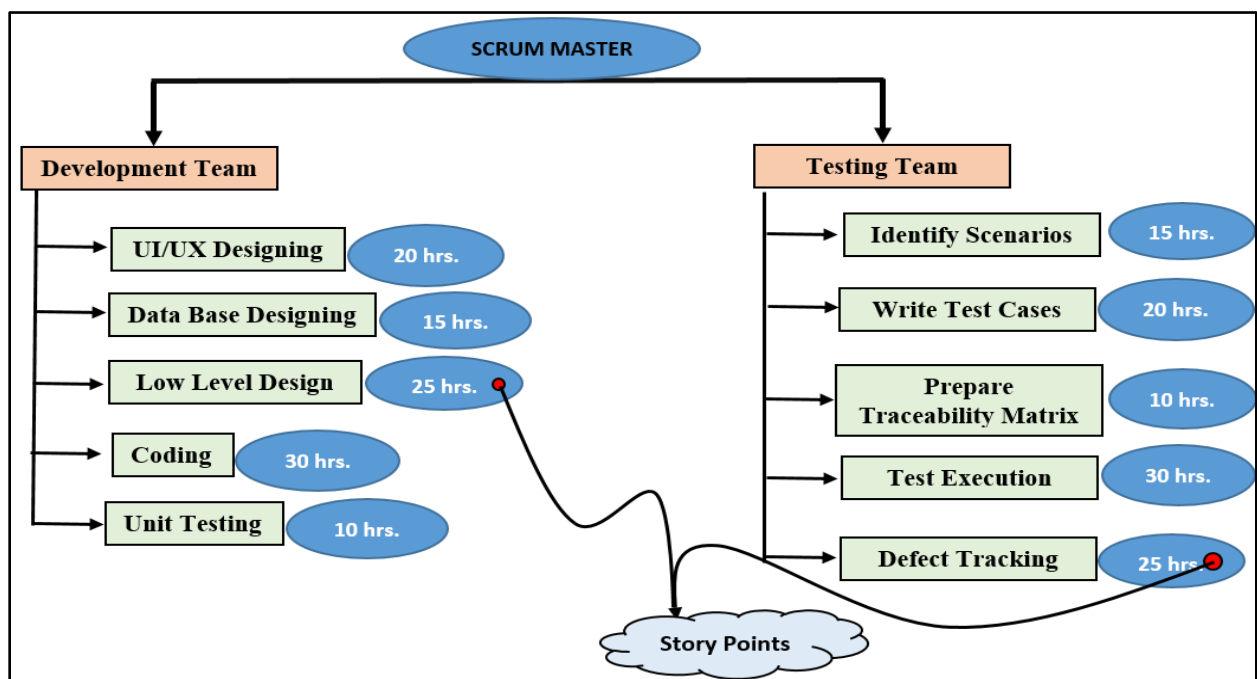
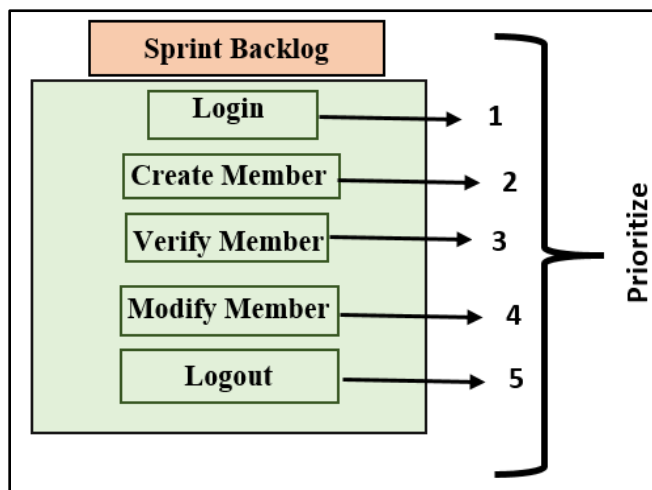
- **BA:** Collects the requirement and converts into SRS document.
- **Software Architect:** Does the HLD and feasibility study of the software technology of the software required or available,
- **UI/ UX designer:** Does the UI/UX (Front end designing)
- **Database Admin:** He manages the database of the application.
- **Network Admin:** He manages the network of setup or environment required to develop and test the application.

2. Formation of Product Backlog:

- It is the prioritized list of stories or requirement that must be developed in complete project.
- Generally, Product Owner, Scrum Master, BA, Architect involved while building product backlog.
- Generally, the stories in the Product Backlog need not to be in detail.



3. Sprint Planning:



After formation of product backlog the scrum team will do sprint planning.

- Scrum team will sit together and pull the stories from the Product Backlog.
- **Sprint Backlog:** It is the list of stories and associated tasks committed by the scrum team that must be delivered within one Sprint. It is committed by Scrum team.
 - Here entire scrum team sit together and pull the stories from product backlog.
 - Scrum Master assigns each story to developers and testers.
 - Each engineer derives the tasks to be completed to build each story,
 - Each engineer estimates the time taken to complete their tasks that is derive story points.
 - Story Points: It is the estimated time for each and every task derived from the assigned stories by the engineers.
 - Generally, sprint Planning meeting should be completed within 2 hours per week and 8 hours in a month.
 - Roles and Responsibilities:
 - **Scrum Master:** This complete meeting is driven by the Scrum Master. His prime role is to facilitate the complete meeting and co-ordinate between the stake holders.

- **Product Owner:** He clarifies all the doubts regarding the requirement. He also set acceptance criteria for each and every requirement.
 - **Development Engineer:** DE should derive the task for building every story.
DE Prioritize which story to be build first and which story to be built at last in the sprint.
DE will prioritize the task.
DE derives the story point.
 - **Test Engineer:** TE will derive the task to be completed for each and every feature or story.
After sprint planning is done actual coding and development will start.
 - **DE Tasks:**
 - UI/UX designing
 - Database designing
 - LLD
 - Coding
 - WBT
 - **TE Tasks:**
 - Identify scenarios
 - Write test case
 - Review test case
 - Execute test case
 - Defect tracking
4. After testing is completed, the product is moved to Acceptance testing where in customer checks whether all the acceptance criteria are met or not.
After the Acceptance testing is successful the product is moved to production.

5. Meetings or Ceremonies in Agile:

A. Stand up/ Daily Scrum/ Daily Roll Call Meeting:

Here entire scrum team meets and every engineer should explain about:

- i. What they have done yesterday?
- ii. What are the impediments or hurdles that they have faced yesterday?
- iii. What are the activities they are planning to do today?
- iv. What are the impediments they are expecting in today's task?
- v. Scrum master tries to solve few problems right there in the meeting, if it takes much time then scrum master note it down in the "**Impediment Backlog**" and solves it later.
- vi. Generally, this meeting will happen in morning and take 10-15 minutes.
- vii. Here everybody should stand up in the meeting so the people will talk to the point.

B. Sprint Planning Meeting:

After formation of product backlog the scrum team will do sprint planning.

- Scrum team will sit together and pull the stories from the Product Backlog.
- Sprint Backlog: It is the list of stories and associated tasks committed by the scrum team that must be delivered within one Sprint. It is committed by Scrum team.
 - Here entire scrum team sit together and pull the stories from product backlog.
 - Scrum Master assigns each story to developers and testers.
 - Each engineer derives the tasks to be completed to build each story

C. Sprint Review Meeting:

Here once product is tested successfully, the demo of the product is given to the customer (PO sometimes) before product is moved to acceptance testing.

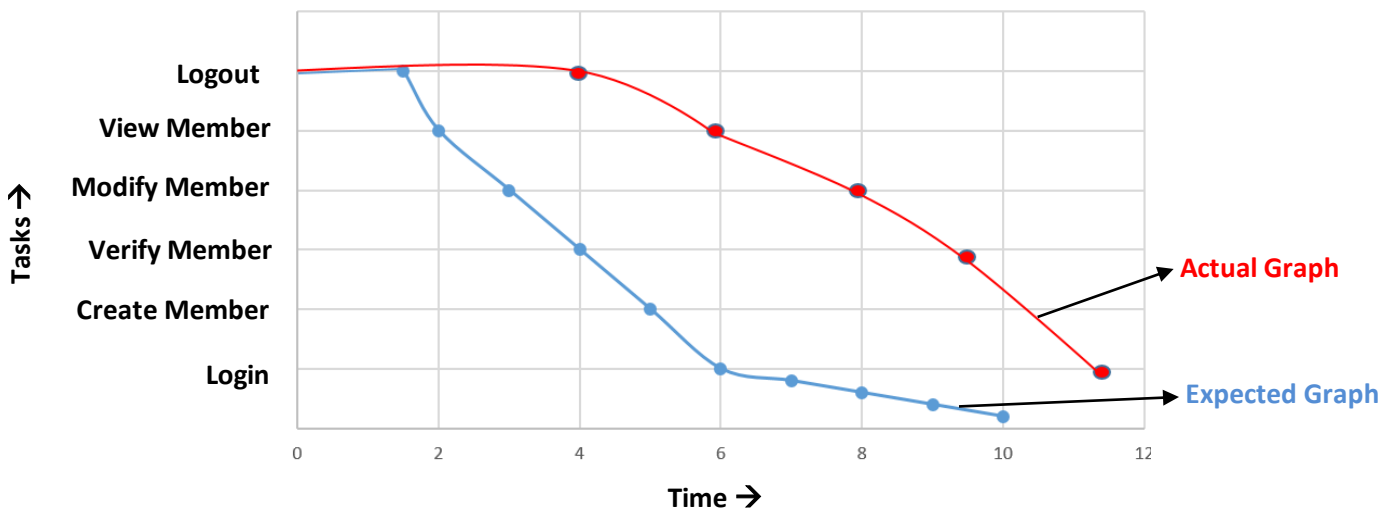
D. Retrospect Meeting:

Here entire scrum team meet and discuss about the achievement (Good process followed) and Mistakes (Wrong activities performed) and we document this document it called as Retrospective Document.

When next release or sprint starts, while doing sprint planning we refer this document and we will plan in such a way that old mistakes will not be repeated and the good activities will be forwarded.

6. Agile daily updates:

a. **Burn down chart:** It is the graphical representation of “Work Left” vs “Time”.



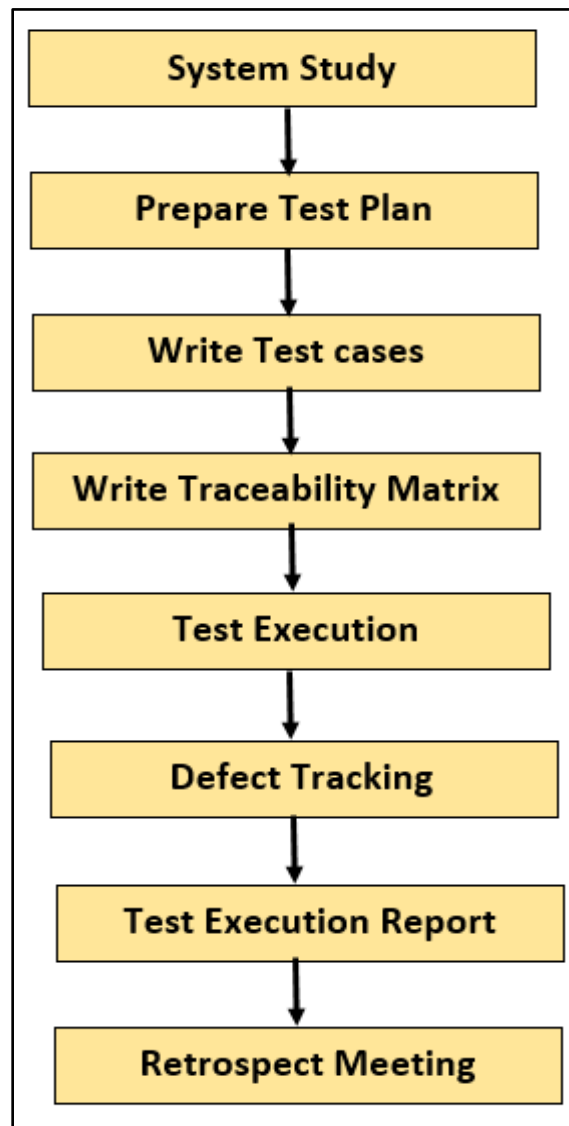
b. **Story Board:** Table which consists of following attributes: Pending Tasks, Task in Progress, Completed Tasks.

Pending Tasks	Tasks in Progress	Completed Tasks
<div><div></div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div></div>
<div><div></div><div></div><div></div><div></div></div>	<div><div></div><div></div></div>	<div><div></div><div></div></div>

Chicken: Chicken is the one into observes and try to understand how sprint is going on and will not be involved in any task.

31) What is STLC and its Process.

- It is step by step procedure or standard procedure to test software.



1. **System Study:** It means we read the requirement and try to understand the requirement. If we have any queries/question we have to interact with the developer's/ business analyst/ customer and get it clarified.
2. **Prepare test plan:** Once we understand the requirement we should prepare test plan. Test plan is a document which drives the future testing activities. This is where we decide:
 - How many engineers are needed to complete the project?
 - What each engineer should do in each stages of testing?
 - What are the types of testing we have to conduct in future?
 - What are the features to be tested and not to be tested?
 - What should be the testing approach?
 - When exactly we should start any activity and end the activity?

3. **Write test cases:** Once planning job is done we start writing the test cases. This activity has got different stages like:
 - Identify all possible scenarios.
 - Write test cases.
 - Test case review
 - Fix the review comments
 - Verify the fix
 - Approve test case.
 - Store test case in the test case repository.
4. **Prepare traceability matrix:** Once test cases are ready the biggest question is what is the guarantee that you have written test cases for all the modules to ensure that we prepare traceability matrix. Traceability matrix is a document which ensures that requirement has got at least one test case.
5. **Test execution:** This is the stage where we test the product for 3-7 cycles, if it is Agile and 40-60 cycles if it is non- Agile model. This is where we conduct all types of testing and we find bugs and the developers will fix the bug to improve the quality of the product. This is where TE are productive to the Organization.
6. **Defect tracking:** We are executing the test cases means definitely we are going to catch lot of bugs. Every bug we are testing to catch should be tracked in very organized way that is called defect tracking.
7. **Prepare test execution report:** At the end of every test cycle we prepare test execution report. It is a document which covers:
 - How many test cases are there?
 - How many test cases are executed?
 - How many test cases are not executed?
 - How many test cases are passes?
 - How many test cases are failed?
 - Pass percentage?
 - Fail percentage?

Note: We send this report to the customer in last test cycles and that is the end of project or release from customer point of view. But from the project point of view we have got one more activity that is Retrospect meeting.

8. **Retrospect meeting:** Here entire testing team meet and discuss about the list of achievements and mistake (good process and wrong process). We document all the mistakes and achievements and this document is called as Retrospect document. In next release or sprint or project at planning stage we open the old retrospect document and plan in such a way that old mistakes are not repeated and good activities are once again adopted. When we do retrospect meeting at the end of the meeting we realize that number of mistake that you have done is very less. That is how we continuously we improve test life cycle.

What are Achievements and Mistake?

Achievements:

- There was a good communication between developer, TE and BA which helped us to complete the testing on time.
- We have conducted Impact Analysis meeting clarifying the doubts in the requirement which helped us to identify impacted areas.
- Developers fixed the blocker defect in early stage itself because of this we are able to complete testing very soon.
- We swapped our modules between TE's and tested the modules which help us to identify more number of defects.

Mistakes:

- BA has delayed in clarifying the doubts in the requirement because of this documentation we got delayed.
- My team didn't review test cases properly because of this test coverage was not good.
- Developer took lot of time to fix the defects because of this it delayed our testing activities.

Difference between verification and validation

Verification:

- It involves in review, walk through and inspection.
- It involves different activities like requirement review, design review, code review, Tc review test plan review.
- Here we ensure that are we building right product.

Validation:

- It involves actual testing.
- It involves different activities like FT, IT, ST, AT, SYT etc.
- Here we ensure that are we building product right.

What are the levels of testing?

Unit testing, Functionality testing, Integration testing, System testing and Acceptance testing.

What is the difference between static and dynamic testing?

Static:

- It is the verification process
- It involves different activities like requirement review, design review, code review, test case review, test plan review
- To do this no need to execute program
- To do this checklist is required.
- We do this prevent defects.

Dynamic:

- It is the validation process
- It involves different activities like FT, IT, ST and AT etc.
- To do this we need to run the program.
- To do this checklist is not required and thus we find the defects.

What is the difference between functional and non-functional?

Functional:

1. Here we check whether the application is working according to functional requirement.
2. Here we check whether feature is working or not.
3. It involves different testing like FT, IT, ST and AT

Non- Functional:

1. Here we check whether the application is working according to non -functional requirement.
2. Here we check whether feature is looking good or not.
3. It involves different testing, performance (load, stress, volume, soak).

What is Alpha Testing and Beta Testing?

Alpha Testing:

- * It is the testing done by Test Engineers before we give product for Acceptance Testing.
- * As being a Test Engineer, we cannot randomly test the software. So in order to make sure that the software is tested and all the acceptance criteria which is given by customer is met we go for Alpha Testing.
- * Alpha Testing is basically done in Product based companies.

Beta Testing:

- * It is the testing done by end-users based on their feedback the product will be released.
- * Beta Testing is usually performed by “Real Users” or “End Users” and it considered as a form of External User Acceptance Testing.
- * Beta Testing reduces product failures and risk and it provides increased quality of the Product through customer validations.

Alpha Testing	Beta Testing
Testers who tested the product will be involved in Alpha testing.	Client or End-Users will be involved in Testing.
Involves both W.B.T and B.B.T.	Involves only B.B.T.
Requires Testing Environment to Test.	The Software is made available to the public and said to be Real Time Environment.
It requires long Execution cycle.	It requires only few weeks of Execution.
Issues or Bugs can be communicated to developers as soon as possible.	Most of the Issues or Bugs is taken as a feedback.
It is done to ensure the quality of the product before we move it to Beta Testing.	It concentrates on the quality of the product and ensures that the product is ready for the Real Time Environment.