SDLC

2-05-2021

(Software Development Life Cycle)

It is a procedure ~~of~~ to develop the software application.

→ It consists of different stages or phases.

CRS : Customer requirement specification

BRS : business " "

SRS : System / software " "

BA : Business Analyst.

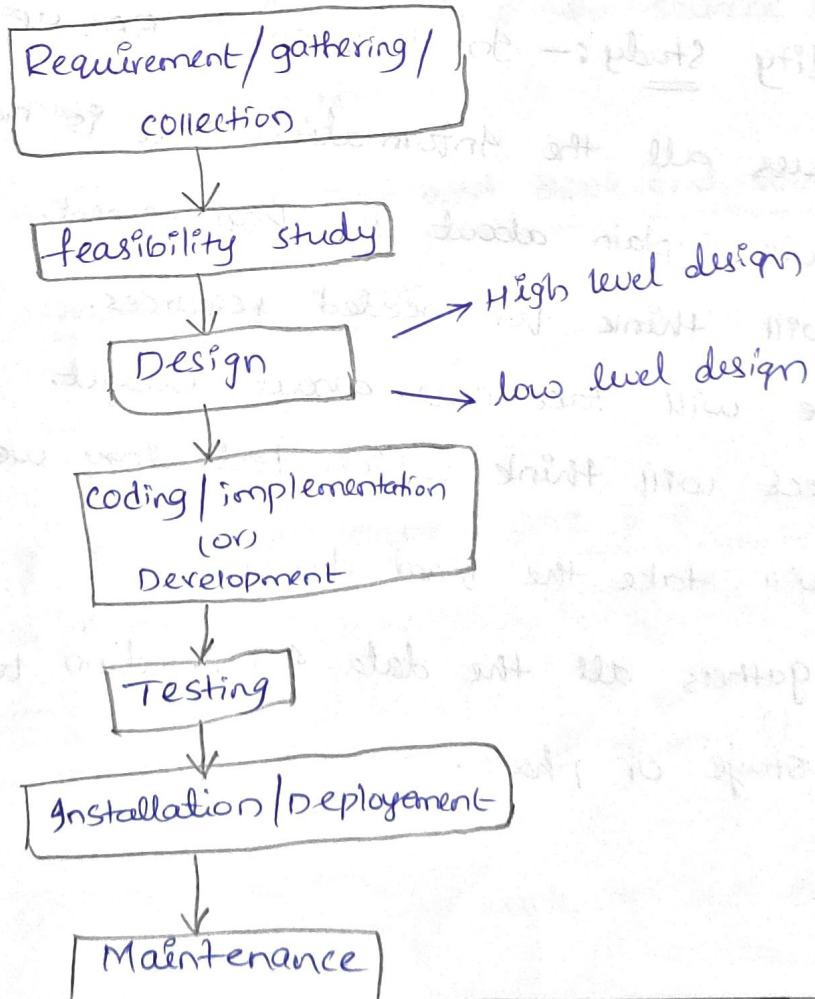
HR : human resource.

HLD : high level design.

LLD : Low " "

TE : test engineer.

PM : project Manager.



→ Requirement / Gathering / collection :-

- Usually customer or client gives a requirement in the form of CRS/BRS. And it is converted into SRS by BA. BA is business analyst. BA acts like bridge b/w customer and Company.

- BA should be good in

→ Communication skills
→ Domain expert
→ Convincing skills
→ Techincal expert.
→ Analyzing skills.

- BA collects all the requirement from the customer and gives to Software Company.

- This is Requirement / collection.

→ Feasibility Study :- In this phase BA, HR, ARCHITECT, FINANCE, PM will discuss all the Information that is needed about the project.

- BA will explain about the Requirement.
- HR will think for needed resources.
- Finance will take care about budget.
- Architect will think which technology we should use for the project.
- PM will take the final decision.

and gathers all the data Information to proceed for the next stage or phase.

→ Design:- Once everything is fine, we go for design after Agreement / sign off b/w customer and company.

- In design phase, we have HLD and LLD. Usually design phase is done by architect or senior developer.

- High level design: It is just like a blue print. It shows the external architecture of the Application.

- Low level design: It shows the Internal architecture of the Application.

- Both HLD and LLD are the Documents.

→ Coding:-
→ Design:- Once the design is done, developers starts writing the Source code for application by looking at design and requirement.

- All developers will involve in writing the source code. ie., Senior, Junior developers.

- Developers will write Front End and Back End source codes.

- This is done is Development server. All developers will be involved in this.

→ Testing:-
Once coding is done, developers will give application to test engineers. All test engineers will start testing the application (UI) and find defects.

- If any defects are found in software application, it again goes back to the developers.

- Developers will fix it and again given back to Test Engineers.

- Like this process continues till Application meets/matches the Requirement.

In Testing Server, Testing will be done.

- Once everything is completed, we will go for next phase called Installation/Deployement.

→ Installation/Deployement :- Making the software application available for end users is called "Installation". i.e., Moving the software from Testing Server to production server. (End users will use it).

- It is done by the software company once testing is completed.
- Customer/client should approve for Installation.
- It can be done by Installation Team/Testing Team/Development Team.

→ Maintenance :-

once s/w is provided to customer or end users, if they face any problem, a support has to be provided, that is Maintenance.

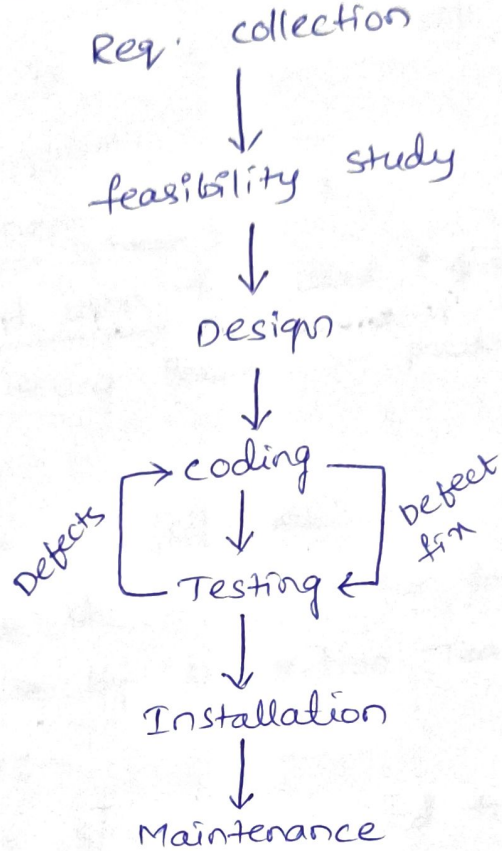
- Initially it will be free, Later it will be paid.
- Free service will be provided based on the agreement b/w customer and company.
- During maintenance, defect fix will be handled and changes will be taken care.
- Changes can be Adding, Modifying, Deleting a feature.
- Maintenance always happens once application is in production Server.

- It will done development/maintenance team (technical team). This is decided by customer and manager.

realme Shot by Akanksha Peddi
2021/07/01 22:15

* Models of SDLC :-

1) Waterfall Model :-



- It is one of the oldest traditional models.

When we go for water fall model?

1) When the customer freezer (doesn't change) the requirement.

2) For any short-term project.

3) For developing simple application.

What are the advantages of waterfall model?

1) we can expect stable application (if requirement doesn't change)

2) There will be no disturbance for the team members if the requirement doesn't change.

Drawbacks of waterfall model?

1) Testing happens only after coding.

2) Requirement and design are not tested.

3) Developers use to do testing before. (currently it is done by test engineers).

4) If requirement changes, it leads to lot of Rework.

Why developers are not involved in testing?

1) Developers always focus how to build the software, not to break the s/w.

2) They always be over confident what they do.

3) They doesn't like to find their own mistake.

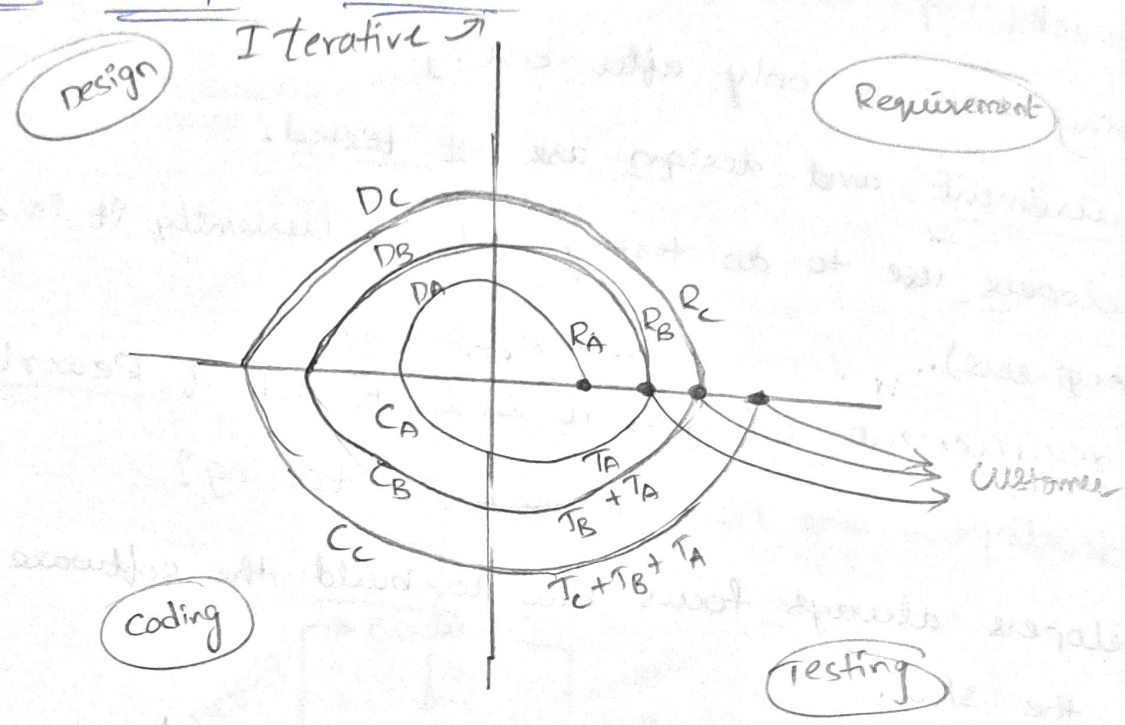
4) Since it is not easy to find their own mistake, even though mistake is there, they will hide it.

→ Why Requirement Changes?

- Due to competition.
- To upgrade Technology.
- To stay in Business.

because of above reasons customers will change the Requirement.

2) Spiral Model / Traditional Model:-



→ This is one of the Traditional model of SDLC.

When we go for Spiral Model?

1) When the customer gives requirement part by part (module by module).

2) Where there is a lot of dependency b/w the modules.

What are the advantages of Spiral model?

1) Customer can see the application partially and can get confidence.

2) Requirement changes can be done.

3) Compare to waterfall model, rework is less.

What are the drawbacks of the spiral model?

1) Requirement and design are not tested.

2) Testing happens only after coding.

3) Developers used to do testing (but now TE is doing testing).

4) If there is any requirement changes it may delay the projects.

Diagram explanation:-

Req - Login - A - Give
Req - compose - B
Req - Inbox - C
Req - sent - D

→ customer will check login module. If he is happy, he will give Req. of compose module.

→ customer check compose and login. If he is happy, he will give Req. of Inbox module.

Requirement changes are of two types.

1) Major:- Whenever there is a major changes we work only on that changes. we can't work on the new requirement.

2) Minor:- Whenever there is a minor changes, we can work on that changes and also we can work on the new requirement.

→ Major:- Ex:-

A - completed (1st cycle)

B - completed (2nd cycle)

Major changes in A - only changes in A will be done. (3rd cycle).

C - start with C (4th cycle).

→ Minor:- Ex:-

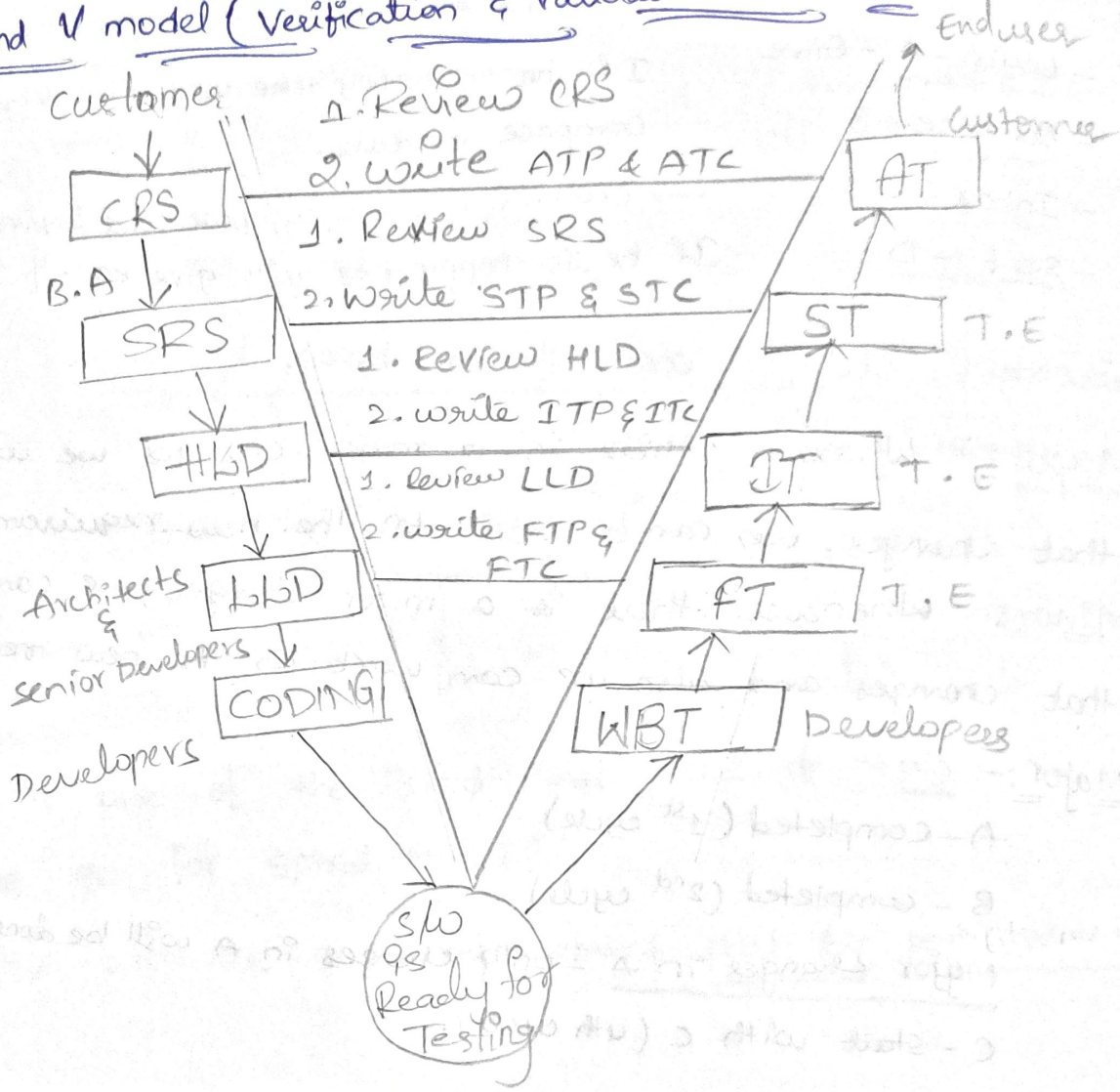
A - completed (1st cycle)

B - completed (2nd cycle)

Minor changes in A & C - will work on minor changes in A & will also work on C module (3rd cycle).

Note:- In case of major changes, cost will ↑ce for customer, so customer has to think before making any changes.

3) V and V model (Verification & Validation model):-



ATP → Acceptance Test plan

ATC → " " case

STP → system Test plan

STC → " " case

ITP → Integration Test plan

ITC → " " case

FPT → Functional Test plan

FTC → " " case

WBT → white box Testing.

V and V means Verification and Validation model.

- Verification is a process of checking "are we building product right?"
- Validation is a process of checking "are we building right product".
- V and V is done only by test engineer.
- Verification is QA process.
- Validation is a QC process.

When do we go for V and V model?

1) When the customer needs high quality products.

2) For complex applications

ex:- Banking, health care applications, space applications, airlines, navy applications.

3) For long term projects (more than one year).

Advantages:-

1) Testing is started at initial stage.

2) Requirement and design are tested.

3) The downward flow of defects are less.

4) Requirements changes can be done.

5) Quality will be high compared to other models.

6) Rework will be less.

Drawbacks:-

1) Documentation work will be more.

2) Too much of resource are needed.

Explain about V and V model?

V and V means Verification and validation model. It is one of the best model in SDLC. In this model the development and testing are done parallely, the left side of model is done by developers and the right side of the model is done by the test engineers.

When the customers give the requirement of 100 pgs documents in the form of CRS. It is converted in the form of SRS by BA. At same time, review of CRS is done by test engineers. If there is any mistakes it goes back, if not it will continue the process. The SRS will be reviewed against the CRS to find the defects. Parallelly they prepare the test plan and test cases.

Once the documentation of development process is done, with the design and coding, the software is ready for testing.

First testing is white box testing. This testing is done by the developers, then there will be FT it is done by the TE. At the same time the execution of the test cases are also done.

After FT the IT is done, later ST and then AT is done by the customer then it is released to end users.

The review of the document is verification and it is based on QA.

The testing of application is validation and it is based on QC.

4) Prototype model:-

Prototype is a dummy model that is a non-working application.

When we go for prototype model?

1) When the customer is not clear about requirement.

2) When the software company is new to the domain, then they will go for prototype model.

3) When the developers are new to the technologies. It is an experimentation.

4) When the customer and software company are new to the business.

Advantages of prototype model

1) Initially customer can get to know what he gets on.

2) Initially itself developers will also come to know what they should deliver on last day.

3) Requirement changes can be done initially itself.

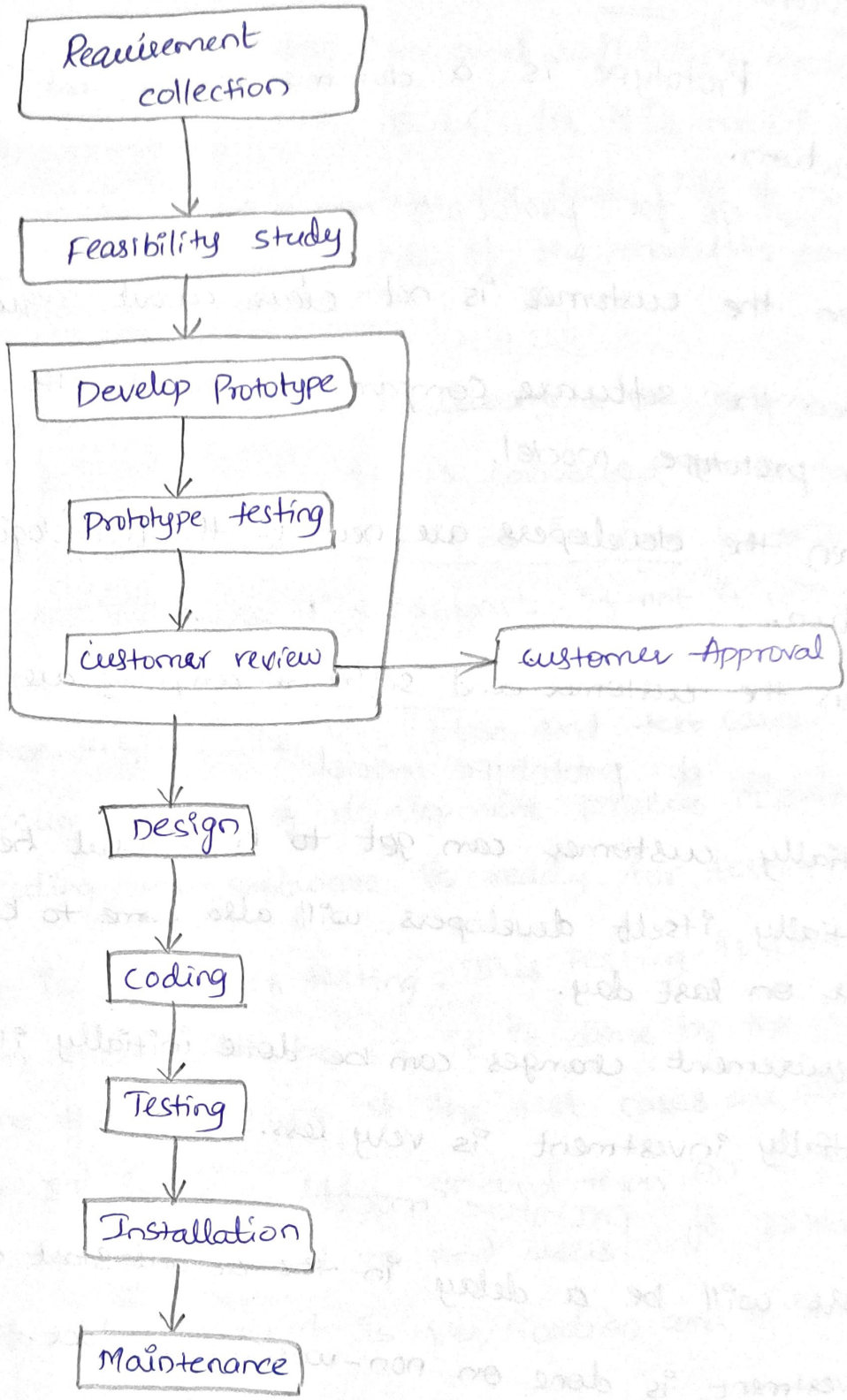
4) Initially investment is very less.

Drawbacks of prototype model

1) There will be a delay in the actual start of the real project.

2) Investment is done on non-working product.

3) Too many changes can disturb the rythm of the company.



Note:- Once i get a job, I may be working on any of these models (or) combination of these models (or) comeup with a new model from these models (or) I don't know which model I am using.

→ When we combine two (or) more models that is hybrid model.

* Technologies used in Software Industry

1) Req :- Requirement

2) Customer/client :- A person who gives the req to the software company for his business needs.

3) BA - Business Analyst :- A person who interacts with customer to understand what exactly they need. BA is a bridge between company and customer. BA converts CRS/BRS into SRS/FRS/FS.

4) CRS - Customer req. specification

BRS - Business req. specification

It is a document given by customer explaining his business needs.

SRS - Software/System req. specification

FRS - Functional req. specification

FS - Functional specification.

It is a document which explains in detail about how a software should look and work along with a diagram.

5) Developer :- A person who writes a source code to develop a software application based on Req.

6) Test Engineer :- A person who test an application which is given by developer and find defects based on Req.

7) End User :- The people who uses the application at last. They are final users of an application. End User can a public person.

8) Defect :- It is a deviation in the Application with respect to requirement. It is formal name.

9) **Bug:-** It's an informal name of a defect.

Whenever a defect is found by a test engineer it should be raised to the developers. Once the developers accept the defect, it is termed as BUG.

10) **Error:-** Mistake in the source code is known as error which is done by Developers.

11) **Issue:-** Problem faced by the customer or the end users.

12) **Failure:-** Multiple issues will lead to failure.

- When the end users face lot of issues that will lead to the failure.

QA and QC:-

QA:- is Quality assurance.

It is a process oriented.

QC:- is a quality control.

It is a product oriented.

QA	QC
1) It is a procedure that focus on providing assurance that quality requested will be achieved.	1) It is a procedure that focus on fulfilling the quality requested.
2) QA aims to prevent the defect.	2) QC aims to identify the defect.
3) It is a method to manage Quality verification.	3) It is a method to verify the quality validation.

4) It doesn't involve executing the programs.	4) It always involves executing the program.
5) It is a preventive technique.	5) It is a corrective technique.

RTM:- requirement traceability matrix

KT:- knowledge transfer.

Domain:- It is a categorization of the s/w application into various fields. Each s/w company will work for various domains.

→ Finacle:- It is a banking software used by the banks.

Examples:-

BFSI - Banking, Financial, services, Insurance (Infosys, TCS, ---etc).

Healthcare - Legato, emids, GE ---etc.

Education - Byju's, Toppr--etc.

Social Media - FB, whatsapp, Instagram--etc

Entertainment/Gaming - Sony, EA sports, Gameloft, Pole to win--etc.

E-commerce - Amazon, Flipkart, Myntra--etc.

Travel and Hospitality - Redbus, Abhibus---etc.

Retail - wells Fargo --etc

Telecom - TCS, ---- etc.

Servers / Environment :-

A server is a computer with high configuration.

- It is like a super computer.
- It accepts the Request from user, identify the user and sends the response to the user.
- All these will happen through Internet and user is recognized from IP address.

we have usually 3 types of servers.

1) Development server. (Used by developers)
2) Testing server (or) QA server (Used by test engineers)
3) Production server (end users)

Eg:- www.dev.facebook.com
www.qa.facebook.com
www.facebook.com

we have 3-different servers because developers, testengineers, end users should use the application and work independently.

- when developers are working for '1' feature. That feature is given to testing team, during that time developers can start working for another feature.
- Due to this

1) Everybody can work independently.
2) Dependency will not be there.
3) If any problems occurs it can be handled separately.
4) Disturbance will not be there for end users.

* Software Testing :-

- Testing the functionality of an application with respect to given Req.
- checking the application with the intent of finding the defect.
- checking the behaviour of an application to see whether it meets customer Req or not.
- Testing the process of QA and QC.

- **Manual Testing :-**

Testing the software without using any tool and ensure its working fine.

- **Automation Testing :-**

Testing the software by using the tool to ensure it is working fine. (the tool can be selenium).

Ex:- TV Remote:- If we operate by using the remote it is automation testing.

- If we do without remote it is manual testing.

- Why testing is Important? Or If testing is not done, what may happen?

Every s/w is developed for business purpose if testing is not done, end users may find the defects while using the applications. This will spread negative impact in the market and no. of users who uses application will be recorded. This will become loss for the investor on business. To avoid all those things s/w testing has to be done before it is released to end users.

Ex:- Andhra bank app

- I send money 10,000 but if I got message as 1000000 transferred successfully. It is one type of defect. It will effect on business.

- Difference b/w Developer and Test Engineer:-

Developer	Test Engineer
1) Build a software.	1) Break a s/w (finding defect)
2) He should be good in logic.	2) He should be good in creative, out of box thinking.
3) +ve thinking.	3) Positive & negative thinking.
4) Aim for construction.	4) Aim for perfection.
5) Single thinking.	5) Multi thinking.
6) Narrow mind	6) Broadmind.

* Senarios :-

Testing the application in all the possible ways is 'Scenarios'.

- It can be +ve scenarios or -ve scenarios

- Testing the application with valid or expected data is called +ve scenarios. It is also called postive testing.

- Testing the application with Invalid or unexpected data is called -ve scenarios. It is also called negative testing.

- To identify the defects we should find the scenarios first.
- Scenario is not a defect or defect is not a scenario.
- +ve scenario and -ve scenarios are not a defect.
- Few cases we may get confused, whether it is a +ve scenario or -ve scenario. In this case we can judge whether it is a defect or not a defect based on Requirement.

1) We can get positive Scenario with a defect
2) " " " " " without a defect
3) " " " Negative " with a defect
4) " " " " " without a defect.

Note :- Unless interviewer also don't divide positive & -ve scenarios.

Examples:-

* Write Scenarios for Manual operating Lift ?

1) verify whether Lift is installed properly
2) verify whether UP arrow button is there outside the lift for all floors.
3) verify whether Down arrow button is there in all floors.
4) Verify by pressing up arrow button
5) verify by pressing down arrow button
6) Verify by pressing both UP and down arrows at a time.
7) verify whether light is appearing once we press the arrow buttons.

8) verify whether Fan is installed inside the lift.

9) verify whether Fan switch is there inside the lift.

10) verify by turning ON the Fan when Fan is OFF.

11) verify by turning ON the fan when fan is ON.

12) verify by turning OFF the fan when fan is ON.

13) verify by turning OFF the Fan when Fan is OFF.

14) verify whether light is installed inside the lift.

15) " " " " switch is there inside the lift.

16) " by turning on the light when light is off.

17) " " " " " " " " " ON.

18) " " " off " " " " " ON.

19) " " " " " " " " " off.

20) " whether we have 10 floors buttons to select.

21) " " every button has different number on it.

22) " by pressing 2nd floor button being in '0' floor (ground).

23) " once we press any floor button, that floor button light should be ON.

24) verify once lift comes to requested floor, it should stop automatically.

25) verify by pressing same floor button from same floor (ex: 0, press 0)

26) verify whether we have STOP button inside the lift.

27) verify by pressing STOP button while lift is moving.

28) verify by pressing STOP button while lift is stopped.

29) verify whether doors are installed inside the lift and also outside the lift.

30) verify lift should not move if the doors are opened.

31) verify we get notification if the lift door is not closed properly.

32) verify whether doors have handle.

33) verify by opening the door while lift is moving.

34) verify by smoking (fire) inside the lift.

35) verify by pouring water on the buttons of the lift.

36) verify by applying sanitizer for the lift and buttons.

37) " " pouring off the electricity while lift is moving.

38) verify whether lift is moving smoothly b/w the floors.

39) " " " " stopping smoothly.

40) " correct floor numbers are displaying in the display

41) " by appling lesser than 500kg load and travel.

42) " by " more than 500kg " " "

43) " " " equal to 500kg " " "

44) " the space inside the lift.

45) " whether we have ventilation inside the lift.

46) " the instructions on the lift.

47) " " Brand name is proper on the lift.

realme Shot by Akanksha Peddi
2021/07/01 22:22

~~Note~~:- Unless

Drawbacks of Manual Testing:-

1) It is a time-consuming process.

2) It is a repetitive in nature (Because of Regression Testing).

3) Turn around time is more in manual testing. So we go for automa-tion testing.

4) Need more Man power.

* Retesting and Regression Testing:-

→ **RETESTING:-**

Testing the defect fixed which is done on an Applica-tion is called "retesting".

→ **REGRESSION TESTING:-**

Testing the impacted areas of an application:

1) After the defect fixed.

2) After changes done in application.

- changes can be adding, modifying and deleting a feature.

- **Why** Regression Testing has to be done?

When defect is fixed or New feature is added, old/existing features might stop working. To ensure existing feature are still working fine, we have to do "REGRESSION TESTING".

Since regression is repeatedly done, many company will go for Automation.

We can also do Retesting & Regression testing manually.

Ex:-1 TEST APPLICATION - TE

- Find defect - (Front camera not working) - Test Engineer.
- Raise defect / Log defect / Report defect - TE to developer.
- Developer will check that defect and Accept defect.
- Fix the defect (modify the source code of front camera) - Developer.
- TE - Test the Front camera again - RETESTING
- Impact Area Testing - REGRESSION TESTING

Impacted areas for this defect to be:

Back camera	Settings / Filters	Instagram camera.
Flash	Quality	
Gallery	pixels	
zoom out	Scanning the QR	
video	Whatsapp camera	

Ex:-2

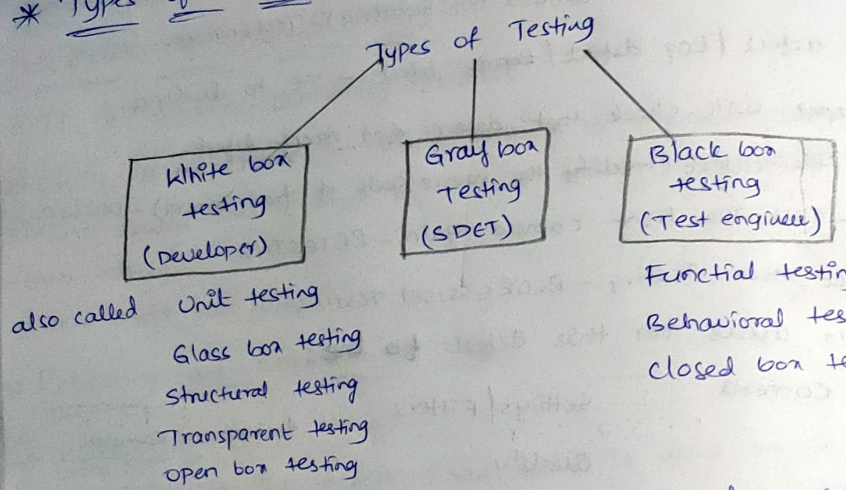
Valid un and valid pwd and click LOGIN - DEFECT (raise) - Fix - RETESTING

Valid un and Invalid pwd and click LOGIN - No defect - Regression

Valid	"	"	Invalid	"	"	"	"	"	"	"	"
Invalid	"	"	Valid	"	"	"	"	"	"	"	"
Invalid	"	"	Invalid	"	"	"	"	"	"	"	"
Valid	"	"	Blank	"	"	"	"	"	"	"	"
Blank	"	"	valid	"	"	"	"	"	"	"	"
Blank	"	"	Blank	"	"	"	"	"	"	"	"

realme Shot by Akanksha Peddi

* Types of Testing :-



→ **White box testing:-** Testing the source code or internal structural of an application is called "white box testing".

When situation demands, even Test Engineer can do this if we know coding.

→ **Black box Testing:-**

Testing the User Interface (UI) or graphical User Interface (GUI) of an application is known as black box testing. It is done by test engineers.

Here we don't worry about internal structure of an application.

→ **Grey box Testing:-**

Testing the source code and user interface of an application parallely, together is called "grey box testing".

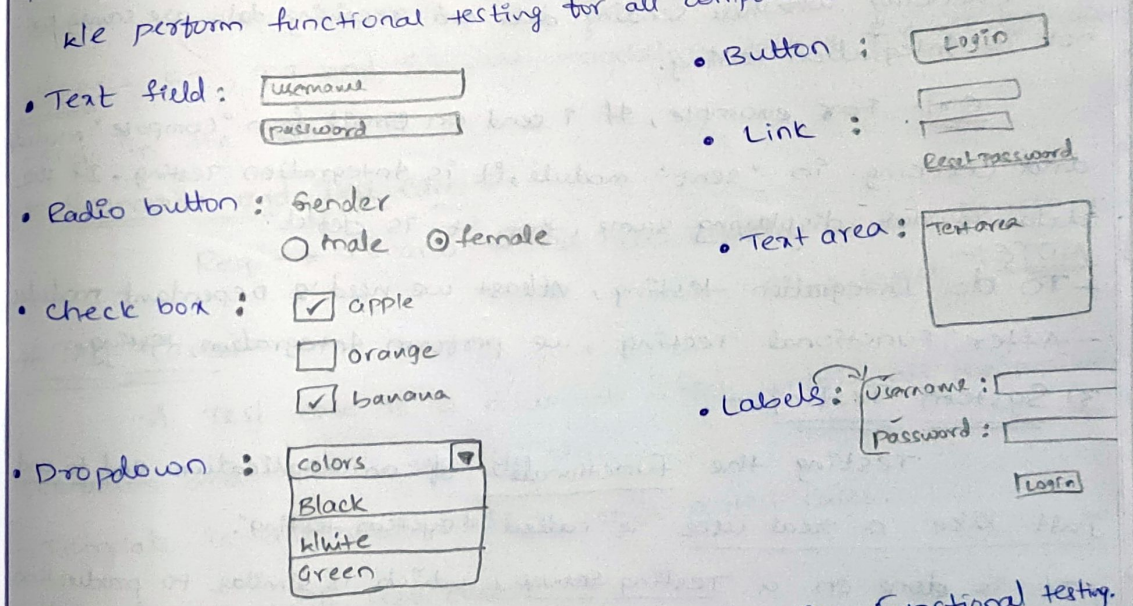
It is done by a person who is aware of both testing and source code of an application. Means, he can be a developer or test engineer OR he can be SDET (S/w Development Engineer in TEST).

- Types of Black box Testing:-

1) **Functional Testing / Field level Testing / Component Testing:-**

Testing each and every component independently and thoroughly with respect to requirement is called "Functional testing".

We perform functional testing for all components like:



- When developer gives one module, we start with Functional testing.
- Before this, we have to be ready with all scenarios of all the modules.
- If developer gives multiple modules, we should do Functional testing for all modules separately.
- Here we should check all positive and negative scenarios for each and every component and find defects.

Ex:- Separately check every module.

Login module, compose module, Inbox module.

2) Integration Testing:-

Testing the data flow b/w two or more dependent modules is called "Integration testing".

Whenever we are testing one module and if we are sending data to other module, Integration will happen here.

checking whether sending data and receiving data are same or not is Integration testing.

Ex:- For example, If I send an Email from "compose" module and checking in "sent" module, it is Integration Testing. If the data is not displaying same, then it is defect.

NOTE:-

- To do Integration testing, atleast we need '2' Dependant modules.
- After Functional Testing, we perform Integration testing.

3) System Testing:-

Testing the functionality of an application end to end just like a real user is called "system testing".

- It is done on a Testing server, which is similar to production server.

- System Testing Scenarios

Ex:-1 Signup → Login → compose → sent items → logout

Ex:2 Login → Inbox → Read email → Reply → sent items → logout

Ex:3 Login → settings → Reset password → Save new password → login (new pwd) → logout.

Note:-

→ we need almost all modules to perform system Testing.

→ After Functional testing and Integration testing, we do system testing.

→ Here we get confidence to release application to customer.

→ Integration testing scenario:-

Ex:- verify whether message sent from compose to visible in sent.

Note:- FT, IT and ST are mandatory testing for all applications in the world.

* Scenario and Test Cases:-

Req → scenario → Test cases writing → execute Test cases → defect.

* Test Cases:-

A Test case is a document which contain all the possible scenarios and explains step by step.

- Template is a format used to write a test cases.

When do we write test cases?

When the developers are developing the application, we will write a test case.

There are two stages

1) Test case preparation/writing

Under the test case preparation, we have to fill following sections like HEADER, BODY, FOOTER Sections.

- In body selection, we will fill till EXPECTED RESULT.

2) Test case execution

After application is given by developer, we will fill Actual Result, Status, Comments.

→ Difference b/w Scenario and Test cases:-

Scenario	Test Cases
1) It will tell all the possible ways we can test the application.	1) It is step by step procedure to test the application.
2) scenarios says 'What to test'.	2) Test cases says 'how to test'.
3) scenario is a high-level-document.	3) Test case is a low-level document
4) scenario doesn't have a navigation steps.	4) Test cases have a navigation steps.
5) scenarios doesn't say where the exact defect is present.	5) Test cases will tell where the exact defect is present.
6) scenarios take less time to write.	6) Test cases take more time to write.

→ In Header Section they are defined as,

→ Test case ID/Name:- It is a unique name/id given for every Test case.

→ Project name:- It is the name of the project which we are working.

→ Req no:- It is the Req number of the test case which we are writing here.

→ Module Name:- It is the name of module which we are testing.

→ Test data:- It is the data which has to be ready before we execute test cases.

→ Preconditions:- It is the condition which has to be satisfied before we start execution.

TEST CASE TEMPLATE:

Header Section

Test case ID/Name:
Project name:
Req No:
Module Name:
Test Date:
Preconditions:
Testing Type:
Priority:
Testing Environment:
Brief Description:

BODY Section

Step Number	Description/Action	Input Data	Expected Result	Actual result	Status	Comments
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						

Footer Section

Author
Reviewer
Approver
Approved date

Testing Type:- It is the type of testing which we are performing here.

→ Priority:- It is the importance of this test case to manage the execution.

→ Testing Environment:- It is the platform where we need to do testing.

→ Brief Description:- It tells what is the aim of this test case.

Note:- 1) Under functional test cases, each and every step is a functional scenario.

2) When we work Integration test cases multiple steps will become one Integration scenario.

3) When we write system test cases all steps will become one system testing scenario.

* TOOLS USED IN INDUSTRIES:-

→ Functional testing tools / Black box testing tools:-

1) Selenium:- It can be used with core java, python, perl-----

- selenium is a tool which is in demand and it is an open source / free tool.

— The demand combination is that selenium is used by core java.

— Thought works is the first company to come with the selenium.

— Selenium is a free and is an open source tool.

2) QTP / UFT :-

(Quick test professional, unified functional test) can be used with VBS (Visual basic scripting, java script).

— Qtp/uft is a paid tool and licensed Tool. It is with HP.

— (Hewlett Packard). Before it was with Mercury.

— QTP is older version, UFT is latest version.

— It is a powerful tool and it is a paid and licenced tool.

— The best combination is that qtp is used by vbs.

→ selenium is in Demand because it is free tool.

→ QTP/UFT is powerful tool.

→ QTP license cost is around 3 to 5 lakhs.

3) winrunner.

4) Silk test.

5) Test Partner.

→ PERFORMANCE TESTING TOOLS:-

Testing an application by applying LOAD and check RESPONSE TIME and stability.

1) Load runner.

2) J meter.

3) QA Load.

4) Silk performer.

5) Neo load.

Load runner is a licensed tool. It is with HP.

→ PERFORMANCE TESTING OR SCALABILITY TESTING:-

Testing the response time of an application by applying load for an application is known as "Performance testing".

Load:- Number of users who uses the application.

Response time:- time taken to get the expected screen based on the user action.

Types of performance testing

- Load testing:- Testing the response time of an application by applying the load which is less than or equal to designed number of users. (it will be given by customer).

Ex:- If the customer want application for 1 lakh users, then the designed number of users is 1 lakh.

- Stress testing:-

Testing the response time of an application by applying the load which is greater than designed no of users, that can be around 10-20% greater than the load.

Ex:- Req - 100000 users → response time should be within 2sec.

Stress testing 1,10,000 users → response time 2sec.

- Volume testing:-

Testing the response time of an application by transferring huge volume of data through application.

Ex:- Huge data sharing through Share it, Bluetooth, google drive by uploading more photos and checking the time taken to transfer the data.

- Soak testing:-

Testing the response time of an application by applying load continuously for some duration of time.

Ex:- 72 hrs more users are using application continuously and we will check the response time for 72 hrs. Apply load for 3 days continuously.

Can we do performance testing manually?

Yes, but there are some drawbacks, like:

1) Too much cost will be involved because of multiple human resource and multiple devices, a huge place for gathering everybody.

2) There will be no accuracy in the results, we human beings we do not perform all the actions at same time. Because of this, result may vary.

So, we go for automation (Ex: J METER, LOAD RUNNER ... etc).

→ Test Management tools:-

1) QC/ALM (Quality center / application life cycle management) — Ex: Samsung.

It is a licensed tool. It is with HP. It is PAID TOOL.

2) JIRA (best tool for AGILE model) — LG

3) OTM (Oracle Test Manager) — Whirlpool

4) Test link — Godrej

Using the above tools we can manage the testing activities like below.

Ex:- QC/ALM. Below are the modules of QC/ALM.

Req - Add requirement / Edit req / Delete req.

Test Plan - Write test cases.

Test Lab – Execute test cases.

Defect – Report / Raise / Log a defect. Track a defect.

What are the test management activities?

1) Going through the req given by the customer and adding to the test management tool.

2) Write test cases by identifying scenarios.

3) Execute test cases and write results.

4) Find defect and report a defect. Track a defect.

→ Defect or Bug Tracking tools

1) Bugzilla :- It is an open source and free.

2) QC/ALM :- It is a licensed tool. It is with hp.

3) JIRA. (partially free)

4) OTM

5) Test Link

– In Bugzilla, we can't add the requirement, write test cases and execute test cases.

– we can only report the defect and track the defect.

Note:-

We can write test cases using excel sheet or tools like QC/ALM, JIRA etc.

Always use the tool if available in the company. If not available, then use template given by manager or create own template.

→ we can Import and Export Test cases between excel and tools.

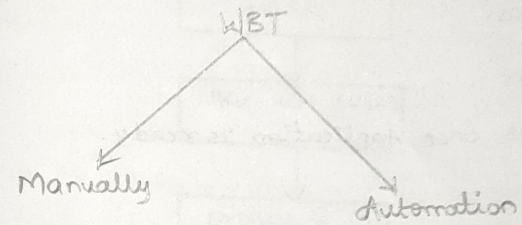
excel (export test case to QC/ALM) → Import test case from excel to tool (QC/alm).

QC/ALM export test cases to excel → Import test cases from qc/alm to excel.

QC/ALM – Excel

Export (Sending) Import (receiving)

* We can do WBT (white box testing) both Manually & Automation.



How to do WBT Manually?

1) Write a program to develop an app based on REQ.

2) Test each and every LOC (Lines of code) without a tool.

3) Check for syntax, Logics etc.

4) Run the developed program and analyse the results.

How to do WBT Automation?

1) Write a program to develop an app based on REQ.

2) Write a testing program to check the developed program using a tool (Ex: JUNIT, NUNIT ... etc).

3) Run the testing program which will run developed program in a tool (JUNIT, NUNIT)

4) Analyse the results.

BBT

Manually — Automation

How to do BBT Manually?

1) Go through Req.

2) Identify Scenarios.

3) Write Test Cases.

4) Execute Test cases once Application is ready.

5) Find defects.

Manually

1) Go through REQ

2) Identify scenarios.

3) Write Test cases.

4) Write a program (Test Script) to Test the application using a tool.
(Ex:- Selenium, QTP/UFT)

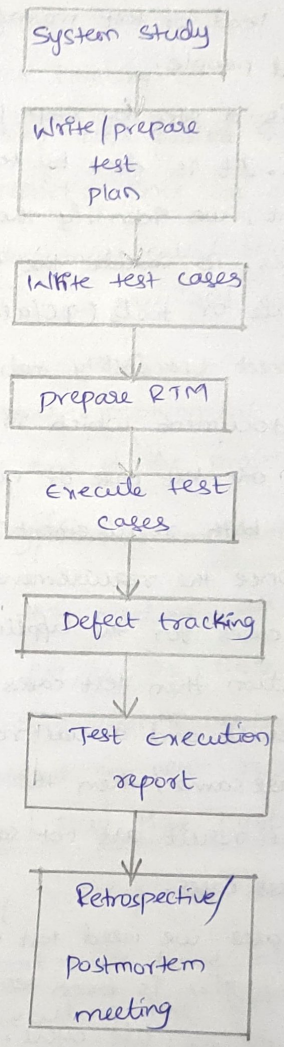
5) Run the Test script in a Tool (Ex: Selenium, QTP/UFT)

6) Find defects.

* Test case - Manually
* Test script - Automation
* what to test / what to tell to selenium - manual testing.
(Scenario)
* Tell to check - Java
* I will check - selenium.

** Software testing life cycle (STLC)

- It is a procedure to test the s/w application.
- It has a different stages or phases like.



→ System study:- It is going through the requirement given by the customer and understand how the system works.

→ Write test plan:- It is a document which is prepared for FUTURE testing activities.

- It is done by the test lead or test manager. Because the plans will be done by experienced people.

→ Write test cases:- It is a step by step procedure to perform the testing on the application. It is done by test engineer. Once we go through the requirement, we identify the scenarios and then converted into the test cases. To write the test case, we need requi-rement and test case template or tools (qc/alm, JIRA).

→ Prepare RTM:- (requirement traceability matrix)

It is a document which is prepared to check whether every requirement has atleast one test case or not.

- To prepare RTM, we need both requirement and test cases.

→ Execute test cases:- Once the requirement is given to the test engineer, he writes test cases for the application. After the developer gives the developed application then test cases are executed and it is compared with expected result and actual result. If the expected result and actual result are same, then the status is PASS, if the expected result and actual result are not same status will be FAIL. This is called execute test cases.

SO, to execute test cases we need test cases and s/w application. This is where exactly the s/w is been tested. This is the most important phase of s/w testing life cycle.

→ Defect tracking:- While executing the test cases, we may come across the defects.

- These defects are raised / logged / reported to the developers.
- once we raise, we should also check what is happening to that defect.
- This is called defect tracking.
- It is done by using tool called Bugzilla, QC/ALM, JIRA.

→ Test execution report:- Once we execute the test cases, we should prepare one document called test execution report. This document tells about how may test cases are PASS and how many test cases are FAIL.

Till here customer involvement will be there. According to the customer, this is the end of the project.

→ Retrospective Meeting/Postmortem Meeting:-

In this meeting, all the team members will gather together like PM, TL, DL, DEV, BA, TM, TE and they discuss about the good and improvements needed once the project is done. This will be very useful for next release or next project.

*** Defect life cycle | Bug life cycle | status of defects :-

→ This is used by raising the defect code.

→ Inside defect report, we have attribute called status.

→ When we are executing the test cases if the expected result and actual result are not same, we came across the defect. Then the defect is raised to the developer, the

Status will be NEW/OPEN.

→ That defect will be assigned to be concerned developer or development lead (if we don't know the correct developer), then the status is ASSIGNED.

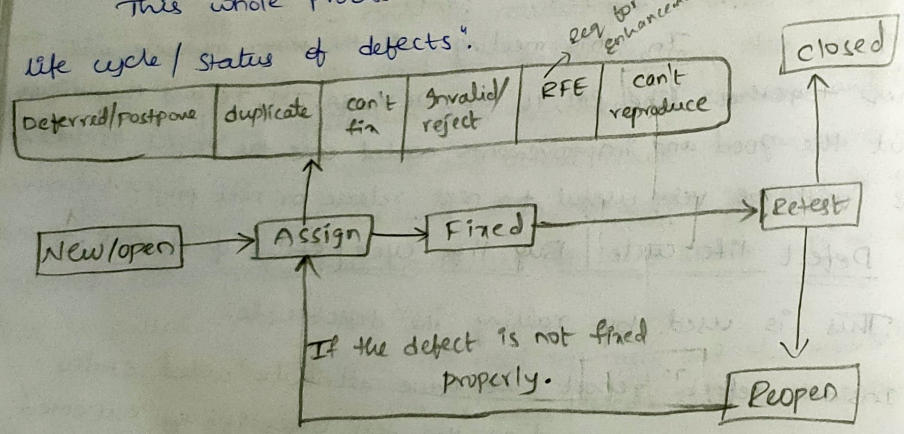
→ Then developer will reproduce the defect and if it is Reproducible, it will be accepted.

→ Then he starts fixing the defect in the development server and it has to be installed to Testing server. Then he should change the status to FIXED.

→ The test engineer starts retesting the defect fixed in the testing server. If the defect is properly fixed, then the status will be CLOSED.

→ If the defect is not properly fixed, then again, the defect is REOPEN to the developer. This process continues till defect goes to closed status.

This whole process is called 'Defect life cycle / Bug life cycle / Status of defects'.



→ New/open, Assign, Retest, closed, Reopen all these status are used by test engineer remaining all are used by developers. i.e., Fixed, postpone, duplicate, can't fix, reject, RFE, can't reproduce

→ Apart from the status which is mentioned above, we also have other status

→ What is duplicate status?

Whenever the test engineer finds the defects and he raises it. If another test engineer or same test engineer finds the same defect and raises it unknowingly/mistakenly, then status of that second defect will be changed as DUPLICATE by Developers.

To avoid this DUPLICATE status,

a) Check whether the same defect is already raised or not in the Repository (Ex: Bugzilla, QC/ALM, JIRA) before raising any defect.

b) Communicate with team members before raising a defect, if you are raising for common module defect.

Note:- Duplicate is a bad status for Test Engineer.

→ What is deferred/postpone status?

Whenever test engineer raises the defect, the developer accepts the defect but he will not fix the defect immediately. He will fix it later, then the status of those defects will be done as deferred/postpone by Developer.

Ex:- Whenever there is major defect and minor defect. Developer have less time to fix the major defect, in this case the developer will fix the major defect and for minor defect he will make status as deferred/postpone.

New/open → Assign → deferred → fixed → closed.

(do retesting before closing)

→ What is Can't reproduce / unable to reproduce status?

When the test engineer finds the defects, but develop-er can't be able to reproduce that defect. So he will change the status as "cannot be reproduced".

Ex:- Mobile hang ... etc.

Reasons for Can't be reproduce

1) Installation problem. (dev server, Testing server)

2) Improper defect report. (chrome, firefox)

3) Inconsistent bug. (A bug which occurs sometimes and doesn't occur sometimes).

Ex:-1) Naughty (DEV), Sweety (TE), ooty, scooty → sweety (saw ghost), video (sweety will take video of ghost) – Naughty (can't reproduce).

2) New/open → Assign → can't reproduce → fixed → closed.

→ What is Invalid / Reject / Not a Defect status?

Whenever the test engineer raise the defect, but developer will not accept the defect as VALID and he changes the status to INVALID / REJECT.

Reasons for invalid:

Due to the misunderstanding of the requirement.

1) If test engineers misunderstand requirement.

New/open → assign → Invalid → closed.

2) If developer misunderstand the requirement.

New/open → assign → Invalid → assign → fixed → closed.

Note:- Invalid is bad for TE.

→ What is Can't be fixed / Can't Fix status?

Whenever the developer is unable to fix the defect that is raised by the test engineer, then he will changes the status as "Can't be fixed". And it is a valid defect, but he can't fix the defect.

Ex:- whenever Technology doesn't support to fix that defect, then developer will change the status to "Can't be fixed".

Finally what to do for this defects will be decided by Managers / customer.

2g - Response time < 2sec - req

5sec - defect

New/open → Assign → Can't fix → closed (after speaking with manager)

→ What is Request for enhancement (RFE)?

Whenever the test engineer raises the defect which is not given the requirement so the developer will take it as suggestion. In this case he will change the status as RFE.

Ex:- Gmail App

Inbox – 1000 emails – delete all emails – (select all is not there) –

- If this is raised as a defect, developer may ~~make~~ it as RFE.
- This should be taken care by BA / Manager for further decision.
- They can speak with customer, If customer wants that feature, it is taken as a new Requirement.

New/open → assign → RFE → closed (Add a comment before closing).

→ **SEVERITY:-**

It will tell how much that defects is affecting the Customer business.

Types of severity

1) Blocker / showstopper
2) Critical
3) Major
4) Minor
5) Trivial (This defect is negligible)

→ Severity is used in defect Report while raising defect.

→ **PRIORITY:-**

Priority says which defect has to be fixed first by the developer.

- For every defect we have to set priority.

Different types of priority

1) High		1) Urgent		1) P1
2) Medium	(OR)	2) High	(OR)	2) P2
3) Low		3) Medium		3) P3
		4) Low		

→ Who will set severity and priority?

Test engineer will set severity and priority.

- Developer or manager can discuss about this if they want to change it, they have to provide proper justification.
- Priority is very much important for the developer to decide which defect has to be fixed first.
- If priority is not there, developer may fix the easy defects first and they leave all the important ~~defects~~ defects.

- As a project point of view, the important part of defects has to be fixed earlier. To manage these things, we have severity and priority for each and every defect.

* Priority can change once other defects are fixed and closed.

Examples:-

1) For Gmail App.

→ Defect id: 101 (login not happening)

Status - New
Severity - Blocker
Priority - High
Assignee - Veerappan
Reported by - dinga

→ Defect id: 102 (inbox mails are not receiving)

Status - New
Severity - Blocker
Priority - High
Assignee - veerappan

→ Defect id: 103 (Cancel button is not working in login page)

Status - New
Severity - Major
Priority - Medium
Assignee - Veerappan

→ Defect id: 104 (when we enter invalid data, login happening)

Status - New
Severity - Blocker
Priority - High
Assignee - Veerappan

→ Defect id: 105 (Cancel spelling is misspelled as cansell)

Status - New
Severity - Minor

Priority - Low

Assigne - Marrappan

→ Defect id : 106 (GMail is shown as Cmail in home page)

Status: New

Severity - Critical

Priority - Medium

2) Give example for Minor severity & High Priority

→ Instead of Gmail, it is cmail

→ Instead of Facebook, it is Phasebook

→ Logos mismatch

→ BATA instead of TATA car

→ Any defect which is high visibility and impacting the brand value, that comes under High priority and minor severity.

→ Instead of Welcome ----- come come

3) Give example for Critical Severity & Low priority?

→ Flipkart APP

Defect - For umbrella, Discount not shown - for all 3 browsers (chrome, firefox, IE).

→ Raise/Report 3 defects.

- Defect : 101 (Discount is not displayed for umbrella in home page in CHROME browser)

Status - New

Severity - Critical

Priority - Medium

- Defect : 102 (Discount is not displayed for Umbrella in home page in FIREFOX browser)

Status - New.

Severity - Critical.

Priority - High.

- Defect : 103 (Discount is not displayed for Umbrella in home page in IE browser)

Status : New

Severity : Critical

Priority : Low

* Real time Testing/Mini Project :-

→ Observations while testing facebook login page :- (Functional testing)

1) I gave (entered) wrong username (errors i deleted last 2 letters of username) and correct password (password) it is able to login.

2) Valid mail id (/ username) and valid password click on create new account then Login will happened.

3) In place of user name in place of 's' i gave '$' and valid password it is able to logged in.

4) If i give in place of +91, -91 i gave them and valid password login will happen.

5) while testing suddenly source code will come.

→ Observations while testing OLA, UBER applications :-

1) When I am trying to book a cab in uber I gave the address details like from and to address is same cab is booked.

2) From hyderabad to Andaman Islands also cabs available it's showing (Uber).

→ Observations while testing Swiggy, zomato apps :-

1) In zomato veg shake is displayed in Red colour instead of green colour (beetroot juice).

2) In swiggy chicken items are displayed in green colour instead of Red colour.

→ observations while testing SWIGGY & ZOMATO applications:-

* Defect Reporting:- 1) Veg shake is displayed in Red colour instead of green colour.

- Defect ID : 001
- Build NO : B01
- Test Case Number: TC_IT_05
- Status: Assigned
- Severity: Major
- Priority: High
- Test Environment: Chrome browser.
- Module Name: ORDER ONLINE (SUPERLEAN MEALS)
- Reported by: Akanksha (TE)
- Assignee/Assign TO: Gouthami (Dev)
- Brief Description: veg shake is displayed in Red colour instead of Green colour (Beetroot juice).
- Test Data: NA (Not Applicable).
- Steps to Reproduce/Description:

1) Open the Browser (chrome)
2) Enter the URL (www.zomato.com)
3) Click on 'Search for Restaurant' field in Home page.
4) Type 'superlean meals' in the search bar.
5) select 'Superlean meals' (any location).
6) Select 'ORDER ONLINE' option.
7) Click on 'SHAKES' option.
8) Scroll down till we get 'Beetroot juice'.

- Expected Result: 'Beetroot Juice' should be displayed in GREEN colour.
- Actual Result: 'Beetroot Juice' is displayed in RED colour.
- Attachment (Screenshot): [ ]

→ Defect Report/ bug report/ raising a defect /logging a defect:-

2) Email sent is not displayed in Sent Items.

- Defect ID : 001
- Build NO : B01
- Test case Number: TC_IT_05
- Status: Assigned
- Severity: Critical
- Priority: High
- Test Environment: Chrome
- Module Name: Sent Items
- Reported By: Soujanya (TE)
- Assignee/Assign TO: Srinivas (Dev)
- Brief Description of defect: Email sent is not displayed in sent Items page.
- Test Data:

Valid Username: qspiders.photo@gmail.com
Valid password: 123456
Sender Email ID: qspiders.photo@gmail.com
Reciever Email ID: veruyadav941@gmail.com.

- Steps to Reproduce/Description:

1) Open the Browser (chrome)
2) Enter the URL (www.gmail.com)

3) Enter Valid Email ID of Sender and click on 'Next' button.

4) Enter valid password of sender and click on 'Next' button.

5) Click on 'Compose' option in the left top of the Screen.

6) Enter valid Email ID of Reciver in 'TO:' text field, Enter all valid Information and click on 'Send' button.

7) click on 'Sent' option on the left side of the Screen.

- Expected Result: Email which is sent should be displayed in sent page.
- Actual Result: Email which is sent is Not displayed in sent page.
- Attachment: Screenshot ☐

3) Login is happening even if we enter Invalid Username.

- Defect ID: 001
- Build NO: B02
- Test case Number: TC_FT_02
- Status: Assign
- Severity: Critical
- Priority: High
- Test Environment: Chrome
- Module Name: Login
- Reported By: Akanksha (TE)
- Assignee/ Assign To: Meenesh (DW)
- Brief Description: Login is happening even if we enter invalid Username.

- Test Data:

Valid Username: Saieshwari12345@gmail.com

Valid password: saieshwari.

Invalid Username: Saieshwari12345,$@gmail.com ($ is extra).

- Steps to Reproduce/Description:

1) Open the Browser (chrome)

2) Enter the URL (www.facebook.com)

3) Enter Invalid Username and valid password and click on 'Login' button. (UN: Saieshwari12345,$@gmail.com, Password: saieshwari).

- Expected Result: Login shouldn't happen. we should get appropriate Error Message.
- Actual Result: Login is successful.
- Attachment (screenshot): ☐

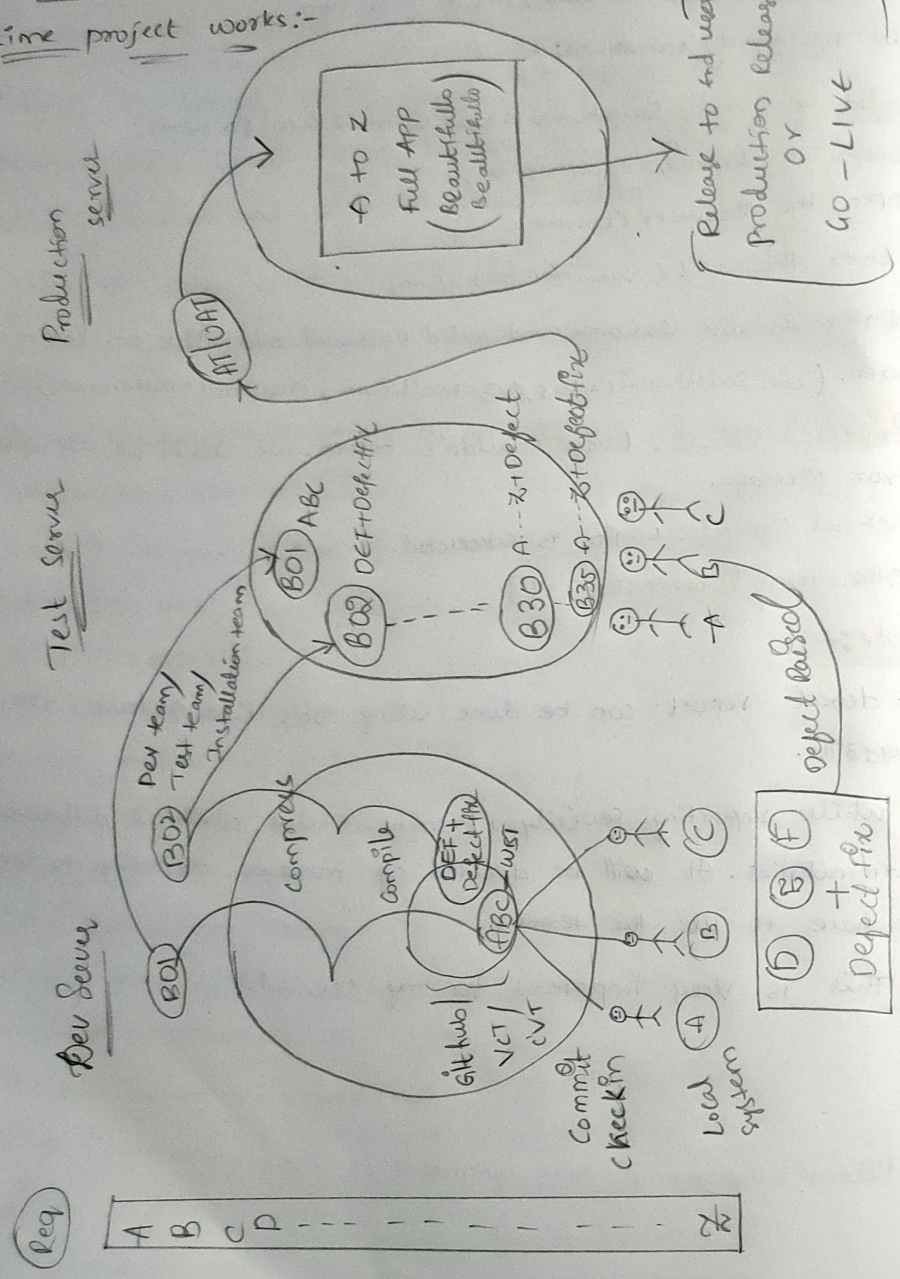
Note:-

1) defect report can be done using tools like qc/alm, Jira, bugzilla.

2) while reporting severity, Priority, status can have different terminologies. It will be decided by manager. According to that we have to use the terms.

3) This is very important for my serviral.

* Build Process / How servers are used in real time / how real time project works:-



Usually we have 3 servers like development server, testing / QA server, Production server.

→ Customer gives the requirement then the developer start writing the source code for developing the application in the local system, this source code will be saved in the repository (Ex: github, VCT, CVT).

→ These white box testing will be done by developers.

→ The source code will be compiled and compressed, then we will be getting a file called 'BUILD'. All this will be happened in development server.

→ The build has to be installed from development server to testing server.

→ It will be done by the installation team or testing team or development team, depending upon project to project.

→ Once build is installed to the testing server, testing team will perform different type of testing depending upon the content of the build.

→ Ex: for new features, FT, IT, will be done for old features regression testing. For defect fixes retesting will be done.

→ While doing the testing, we find the defects that will be raised to the developers.

→ Development team will reproduce the defect and they will fix it in the local system and it is saved in the repository again.

→ Same process will be repeated for every build, until we get the final stable build.

→ Once the testing is completed, we will give the application to

- the customer for acceptance testing.

→ Customer will do acceptance testing in the testing in the testing server or customer will have their own server called user acceptance testing server (UAT server).

→ Once everything is fine application is released to end users that is for production server. This is called Production release or Go-Live.

What is release?

Starting from gathering the requirement, developing the application, testing the application, finally releasing it to the end users is called "one release."

That is called production release or Go-Live of an app.

What is build?

The compiled and compressed format of source code is called "build".

- Build can also be called as "different versions of an application"
- A build contains different formats like compress and archive.
- Compress format:

1). zip

2) Multiple files can be made as single file.

3) Size of the file will be reduced.

- Archive format.

1) . Jar (java archive), .war (web archive), .tar (tape archive),

. gz (Gun zip).

2) Multiple files are made as a single file.

3) Size of the file will be almost same.

Note: Maximum in a Company, build will be in Archive format.

What does build contain?

A build contains

1) New features

2) Old features

3) Defect fixes.

It depends upon the contents which developer will add inside a build, and it varies from build to build.

Note: Initial build is known as Unstable build and final build is known as "Stable build".

What is test cycle?

It is effort or time spent to test the application once the build is given. The duration of each test cycle can be days or weeks or months, depending upon the build and application size.

→ Retesting and regression testing with respect to build and releases:-

A single Application can have single Release or Multiple release.

- A Release can have multiple Builds.
- the final stable build will be released to production server.
- the first build of first release, we cannot do retesting and

- regression testing (Because defect fix is not there and old features are not there).

- For all other remaining builds, we can do Retesting and Regressing testing based on the Situations.

* SMOKE TESTING (OR) BUILD VERIFICATION TESTING (OR) CONFIDENCE TESTING :-

It is testing the basic/critical features of an application before we do through testing like functional, integr-ation, System testing.

Why we do smoke testing?

→ Test engineer will get confidence on basic features is worki-ng fine or not.

→ If we find the defect in the basic feature only, that create a good impact on the testing team.

→ If there are any blocker or Critical defect found in the basic features, developer will get more time to fix the defect.

When we do smoke testing?

→ Once the build is given by the developer, we start doing Smoke testing first.

→ For every build given by the developer we start with smoke testing first. we assure basic feature are working fine.

→ when time is less for testing team to do testing, we will do smoke testing and if customer is fine we will release to customer.

Examples :- What will you test as a part of smoke testing for below applications.

1) Mobile phone :-

Smoke	non Smoke
1) calls	1) Screenshot
2) contacts	2)
3) Messages	
4) speakers	
5) camera	

2)

3)

4)

→ Differences b/w smoke testing & sanity testing.

Smoke testing	sanity testing
1) It is testing the basic/critical features only.	1) It is testing the new features and bug fixes.
2) It is shallow and wide testing.	2) It is narrow and deep testing.

3) It is done on the unstable builds.	3) It is done on the stable builds.
4) Only +ve testing is done.	4) Both +ve & -ve testing are done.
5) It is done on initial builds.	5) It is done on Last builds.
	6) It is the subset of regression testing

Note:- In smoke testing

1) we check only positive scenarios.

2) we check only basic / critical features.

3) From FTC, ITC, STC, take only positive scenarios from all basic modules and create a new test case document called smoke Test case.

4) Since we do smoke testing for every module, we can go for automation.

5) Both smoke and ~~Sccarity~~ sanity are partial testing.

→ Differences b/w formal and informal smoke testing :-

Formal	Informal
1) we have documentation (Test case)	1) There is no documentation.
2) we have a proof.	2) There is no proof.
3) It has a procedure.	3) There is no proper procedure.

* ADHOC TESTING (OR) MONKEY TESTING (OR) GORILLA TESTING:-

Testing the application randomly without following any formal documents like requirement, test cases etc.

Why we do adhoc testing?

An application can be used by different set of people or end users, like :

1) children
2) A person with happy mode
3) A person with depressed mode.
4) A drunken person
5) Illiterate
6) Mad

When we are not able to get the defect through our regular testing like FT, IT, ST we can think about adhoc testing so that we can get more defects.

→ This is the best testing for gaming application.

→ Because while plying games usually children will click where ever they want without their knowledge.

→ This testing is mandatory for gaming applications, for other applications we will do testing based on customer request or based on situation.

→ when we are not getting much defects with functional, integ-ration and system testing think about adhoc testing we will definitely get defects.

→ All the defects we found through adhoc testing need not be fixed by the developer unless it is crashing the application.

Ex:- Adhoc testing Scenarios.

1) Login to gmail from chrome, login to gmail from firefox. Logout from firefox, continue using chrome.

2) Login to gmail from chrome, login to gmail from firefox. Change password from firefox, continue using chrome.

3) Instead of left click, do right click for buttons.

4) In notepad, press space bar continuously for 3 to 4 hrs.

5) Press all buttons at the same time (Login and Cancel).

→ ROADRASH (adhoc testing scenarios)

1) Verify by riding in a wrong direction.

2) " " hitting a board near the road.

3) " " hitting a tree.

4) " " trying to cross a bridge.

5) " " trying to enter water.

6) " " doing nothing for more hours (3)

7) " " climbing the mountain.

8) " " pressing all arrows at the same time.

→ What are the things we expect in gaming application?

- Visuals should be excellent (Graphics).
- Audio should be very good.
- synchronize of audio and video.
- Control options (settings).
- opponents (multi player)
- challenge (Level 1, Level 2, Level 3)
- Scores
- Response (performance)
- Entertainment.

Note:-

- In adhoc testing, maximum we do negative scenarios.
- After smoke testing, FT, IT, ST, we think about adhoc testing.
- Game testing will be done through manual testing usually. Because graphical content cannot be automated.

* Exploratory Testing:-

Whenever application is given, but there is no Requirement for it, then we go for Exploratory Testing.

- Here we find scenarios based on our understanding, convert them into Test cases.

Q) What are the drawbacks of Exploratory testing?

a) It will lead to LOT of miss understanding.

b) It may lead to fight between Developer and Test Engineer.

c) we may understand a Defect as a feature or a feature as a Defect.

d) Because of NO clarity, we may waste lot of time.

Q) How to overcome the misunderstanding?

a) Communicate closely and politely with all team members. (developers, BA, Manager).

Note:- Leave Ego and lets go and work along with team members THIS FOR MY SURVIVAL.

* Compatibility Testing:-

Testing the application with different hardware and software platforms is called "Compatibility Testing".

Why do we do compatibility testing?

1) To ensure that application is working for multiple platforms because there might be different types of users.

2) To check whether the application is consistently working in all platforms or not.

Types of Compatibility:-

1) Software Compatibility testing:-

Testing an application in different software platforms is S/w compatibility testing.

Ex:- In Different OS like:

Windows (w7, w8, w10 .. etc)

Linux (ubunto, federo. etc)

Mac (Macintosh) (Tiger, Lion, leopard, mountain lion. etc).

all above for PC

- For Android (mobile)

Alpa, beta, oreo, pie, kitkat, icecream, eclairs, lollypop, sandwich, foreo, gingerbread, honey comb .... etc.

2) Hardward Compatibility testing:-

Testing an application in different Hardware platforms.

Ex:- In different processor like - Intel, AMD

" " Mother Board like - Intel, ASUS

In different RAM, ROM - Random access memory, Read only memory

In different Hard disk - bits, byte, kb, mb, gb, tb, zb (sony, segate)

In different Graphics card - nvidia.

3) Mobile Compatibility Testing:-

When we test an application for different operating system and also for each different versions like ios, kitkat, windows and for each brand of mobile.

Ex:- Samsung, Realme, redmi, nokia .. etc.

- For each brand, for different models like A series, M series, Galaxy for samsung

Xperia cu, C5, Z5 for SONY

V15, V7, V19 for VIVO.

4) Browser Compatibility Testing:-

Testing an application in different browsers.

Ex:- UC browser, chrome, Mozilla firefox, opera, Safari, Internet explorer, Duckduckgo, Netscape navigator etc

What kind of bugs we find in compatibility testing?

1) Look and feel changes (font and colour changes)

2) Object overlapping (Login and cancel button sitting on one another)

3) Certain images will not display in certain browsers.

4) Scroll bar issues (horizontal & vertical scroll bar may work in one browser and may not work in other browser.)

5) Alignment issues/ problems.

6) Scattered content (cleanliness)

7) Certain buttons, links and components may work in one browser and may not work in another browsers.

→ Major Project:- (Manual testing Project):-

Perform different types of testing like functional, Integration, System, Smoke, Adhoc, Exploratory, Compatibility --etc on real/diff. Applications.

Ex:- Applications and Browsers which are considered for Testing.

1) zoomcar.com

2) Magicbrics.com

3) Bluestone.com

4) Patanjali.com

5) Any application of your choice.

Test these applications in different browsers like Chrome, IE, opera, firefox, uc (any 3 browsers).

- Create one doc (word doc) of all the defects which is found.

Note:- Do this project in PC preferrably.

→ Types of Applications:-

Generally, we have three types of Applications.

1) Standalone applications

2) Client server applications

3) web applications.

1) Standalone applications:-

This kind of applications can be installed, accessed and used without any dependency on server or internet.

Ex:- notepad, calculator, word, calender etc.

- This is the fatest applications with respect to response time.

2) Client server applications:-

In this case we have software in two categories.

a) Client software

b) server software

→ Client software will be installed by the end users usually from play store.

→ Server software will be available at the company location.

→ To use client software to interacting with each other, we need server software. To connect to the software, we need internet.

→ In client server application static elements will be already stored in the mobile, only dynamic elements will be accessed through the internet.

→ This is the fastest compared to the web application and slower compared to standalone application.

Ex:- Whatsapp, gmail, facebook, redbus app.

3) Web application:-

In this we access the application through the browser. Here static and dynamic element will load together.

Ex:- we can access gmail or fb any other application by entering the url into the browser.

- we need internet connection compulsory.
- Here the browser will act as client.

* Globalization Testing:-

Testing an application which is developed for multiple languages is globalization testing.

When the language is changed, the content may not be proper because a machine translator couldn't understand the exact meaning of the word displayed. This may happen because of below reasons.

1) Machine doesn't have feeling.
2) It can't understand exact meaning.
3) " " " " grammer.
4) " " " " spelling.

All these are reasons for wrong translation. So, we go for human translator.

→ If a person is very good with multiple languages local and international, we could perform good globalization testing.

- Japanese, chinese, german, French, Korean, Spanish languages are in high demand.

Ex:- 1) Yahoo -

welcome to Yahoo - English baari raa to Yahoo - Defect.

Yahooge swagatha - kannada

Yahooki swagatham - telugu

2) I Love you - English

Nenu prema ninnu - telugu - Defect

Nenu ninnu premistunnanu - telugu - correct

→ Types of globalization testing:-

1) I18N : Internationalization

2) L10N : Localization

1) Internationalization Testing:-

Testing the application whether it displays the right content at the right place in the right language is called I18N testing.

Ex:- When we are testing an application for English, Hindi, Kannada, Telgu etc, sentences should display from left to right. (i.e. left aligned) whereas for Arabic, Urdu, sentenses should dis-play from right to left (i.e., right aligned).

2) Localization:-

Testing the application with respect to the local culture or local standard to that country or state or region

is localization testing.

Ex:- FOR INDIA — FOR US

RS — $

Dd/mm/yyyy — mm/dd/yyyy

Pin code — zip code

(560097) — (CA 12345), (NY 25634), (SI 58746).

* Rediability Testing:-

Testing the functionality of an application continuously for long duration of time. This can be done by single users. It is to check the consistency of an application.

Ex:- Mobile should function properly with one user for long time.

→ ICICI more reliable than SBI.

* Usability Testing:-

Testing the application to check the user friendliness of it is Usability Testing.

(or)

Checking whether application is easy to use and it gives what we want within less actions.

→ Testing the application, whether it is user friendly or not.

Who should do usability testing?

1) End users is the best.

2) customer or client 2nd best.

3) others project team members.

4) Nobody is there, test engineer has to do usability testing.

Examples:-

1) GPS - global positioning system

VPS - Virtual positioning system - more user friendly.

2) Touch screen - more user friendly

keypad.

3) OLA

Pickup, drop, type of cab, confirm

your name, mother name, father name, age, adharcard (not user friendly questions).

UBER

Pickup, drop, type of cab, confirm.

B.Tech - marks, 10th-marks, yop, backlog! (not user friendly)

4) Mobile

compare to Long press → password → pin → Pattern → fingerprint → facelock → iris → voice. --- etc

(password is user friendly.)

* Accessibility Testing or ADA Testing or 508 ACT rule Testing:-

(American Disability Act)

Testing whether the application is suitable for physically challenged people or not.

Ex:- 1) RGB (Red Green Blue) colour should not be used in the application.

RGB color, color blind people will not be able to view that.

2) Every feature or component in an application should be accessed by

mouse and also by keyboard.

Mouse and keyboard : anybody who is injured with wrist, they can still use the application using the keyboard.

→ Tools used for accessibility testing

1) In focus

2) Wave.

* Recovery Testing:-

Testing the application whether it is able to recover from crash state or not.

Example:-

Intentionally use multiple applications at the same time continuously and make the application CRASH (not responding).

Later close the application and open it again and see whether all previous data is still available or not.

This is RECOVERY testing.

If all data are available and application is working normally after crash, then it is not a defect.

If any data is lost or application is not working properly after crash, then it is a defect.

* AESTHETIC TESTING:-

Testing the beauty of an application is called "aesthetic Testing".

Ex:- Testing colour combinations, font size, font style, image --etc,

* Security Testing:-

It is to check whether all the DATA and RESOURCE-s of an application is accessible only for authorized users and it is highly secured.

Types of Security Testing:-

1) Vulnerability Scanning:-

It is identifying all the risk areas of an application with respect to security.

2) Security Scanning:-

Once the risk areas are identified, we have to secure them with additional safety measures, that is security scanning.

3) Penetration testing:-

Once the security is implemented on the risk areas, check again whether still any risk or data leakage is happening, this is Penetration testing.

4) Risk Assessment:-

Once the security risks are identified, Categorize them into High, Medium, Low and take precautions as needed.

5) Security Auditing:-

This is an internal inspection of Applications and OS for security flaws. An audit can also be done by line by line inspection of code.

6) Ethical hacking:-

It is a process of attacking a system by an internal employee to see whether security measures which is taken is really worth or not.

7) Posture Assessment:-

This combines security scanning, Ethical hacking and Risk Assessments to show an overall security posture of an Organization.

Ex:-

1) http - hyper text transfer protocol

https - " " " " secured - this is secured.

2) OTP, CAPTCHA login are more secured compared to other Logins.

3) Virtual keyboard are more secured than physical keyword.

4) Incognito is more secured compared to Normal browser window.

→ These are more secured compared other applications.

Security Testing Scenarios:-

1) From Gmail Inbox page, Logout from application. click on back button.

If Inbox page appears, that is a defect.

2) Being in Inbox, copy the URL of Inbox page, Logout from application. Paste the URL and see which page appears.

If Inbox page appears, that is a defect.

* FUZZ TESTING or FUZZING:-

It is a testing where we enter invalid or unexpected data randomly to check how system behaves and handles it. (This is a sort of ADHOC testing).

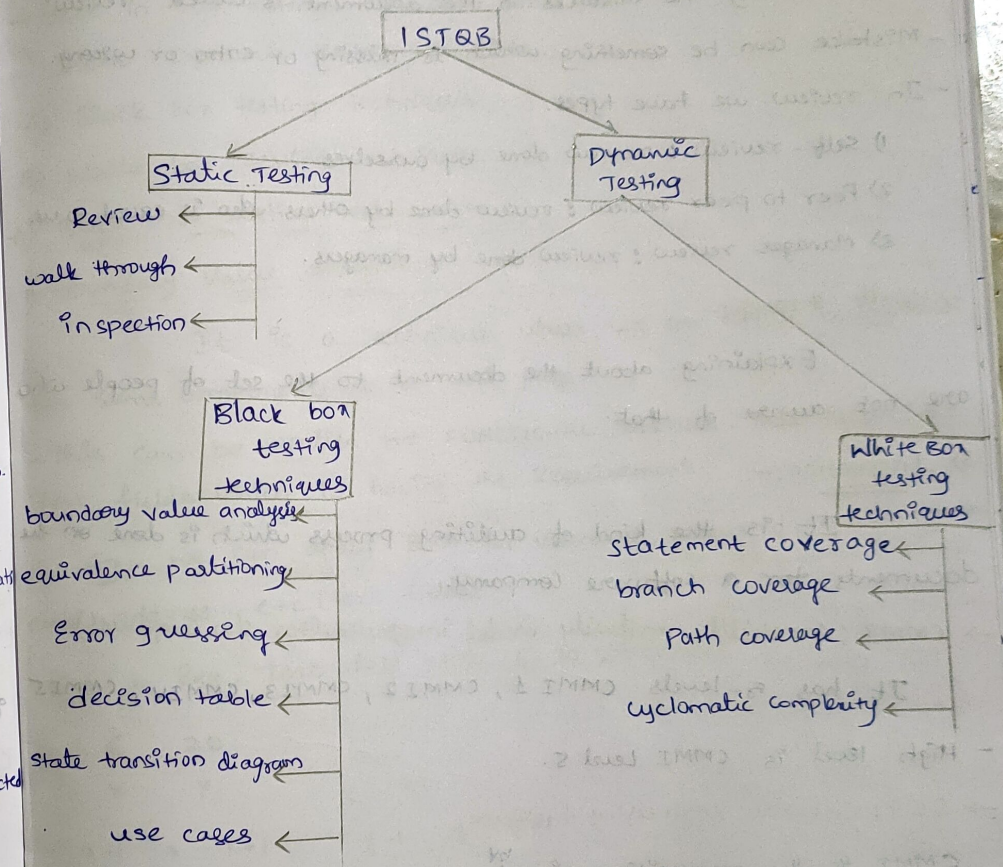
It is also to check the vulnerability of the system. (This is sort of security Testing).

— overall, FUZZ testing is almost a combination of ADHOC and SECURITY testing where we perform through AUTOMATION.

→ Adhoc + Security → Performed using Automation = FUZZ TESTING.

* International software testing qualifications board (ISTQB):-

It is a Certification for software testing Engineers.



I) Static Testing:-

Testing which is performed on the document is called "Static Testing".

→ There is no execution here.

→ Checking req, test case, rtm documents is STATIC TESTING.

Types of static testing:-

1) Review:-

Finding the mistakes in the document is called "review".

- Mistake can be something which is missing or extra or wrong.
- In review we have types.

1) Self-review : review done by ourselves

2) Peer to peer review : review done by others who is equal to us.

3) Manager review : review done by managers.

2) Walk through:-

Explaining about the document to the set of people who are not aware of that.

3) Inspection:-

It is the kind of auditing process which is done on the documents for a software company.

→ CMMI : capability maturity model integrated. (for s/w companies)

It has 5-levels CMMI 1, CMMI 2, CMMI 3, CMMI 4, CMMI 5

- High level is CMMI Level 5.

→ SEI : Software Engineering Institute.

CMMI is the recognition given by SEI

→ ISO : International Organization for Standardization.

It is for Hardware.

II) Dynamic Testing:-

Testing which is done on the application is called "dynamic testing", that is on source code or user interface of an application.

- Here in this testing, execution takes place.
- Under Dynamic testing we have,

A) Black box testing techniques.

B) White box testing techniques.

A) Black box testing techniques:-

1) Boundary Value Analysis:-

It is a technique where we can apply boundaries for some components in a web page.

- This can be applied for FUNCTIONAL TESTING. That is for the Text fields which is having the Requirement.

Ex:- we can apply BVA for username, password, mobile Number, email address ..etc (text fields).

Ex:- 1 Req: Text field with 6-20 characters

6 20

x y

x-1	x	x+1	y-1	y	y+1	
5	6	7	19	20	21	(3 value BVA)
−	+	+	+	+	−	

5	6	20	21	(2 value BVA)
−	+	+	−	

2) Req: Amount Text field with 100 - 500 members.

99 100 101 499 500 501 (3 value BVA)

99 100 500 501 (2 value BVA)

3) Req: 50 to 100 text field

49 50 51 99 100 101 (3 value)

49 50 100 101 (2 value)

49.9 50 50.1 99.9 100 100.1 (Decimal Value)

49.99 50 50.01 99.99 100 100.01 (Decimal value)

4) Req: Phone number field should accept 10 digits only

BVA: 9 10 11

\- + -

2) Equivalence Partitioning / Equivalence class:-

It is the technique where we come up with given range of requirement. Here it contains one valid and two invalid data.

- This can be applied for FUNCTIONAL TESTING. That is for the TEXT FIELDS which is having the Requirement in range.

Ex:- 1) Req: 5 - 10 - 20

2 7 15 30 - Equivalence partitioning

2 7 15 25

\- + + -

2) Req: Text field should accept b/w 6 to 10 characters

6 - 10

2 8 20

Invalid valid Invalid

3) Error Guessing:-

It is the technique where we come with all possible scenarios where we should get Error message.

- Here we have more no. of negative scenarios.
- What is not covered in BVA and equivalence partitioning that we should cover here.

Ex:1 Req: Text field should accept b/w 5-20 (Alphabets only). It is Mandatory.

123456 - We should get Error message.

@#$%^& - " " " " "

abcds @ - " " " " "

Blank (Empty - " " " " "

Note:-

1) once I get a job, I need to apply BVA, EP & Error guessing while writing Functional test cases and while writing Functional test cases and while doing Functional testing where ever possible.

2) These 3 techniques are called as Testcase Design Techniques.

4) Decision table :-

It is the table which helps us to decide how many number of test cases are needed for a given set of Requirements.

- Any complicated requirement can be converted into decision table for easy understanding.

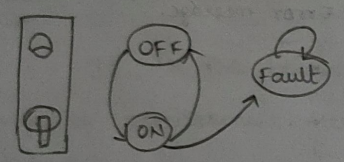
Ex:- ISTQB question paper 1 (Q.No 10).

5) State transition diagram :-

It is a technique which is used to come up with the flow of an application in a easy manner.

- This can be used in system testing and integration testing.
- For atm machine this diagram helps a lot.

Ex:-



→ Q:- Consider the above state transition diagram of a switch. which of the following represents an invalid state transition?

a) OFF to ON

b) ON to OFF

✓c) FAULT to ON

6) Use Cases :-

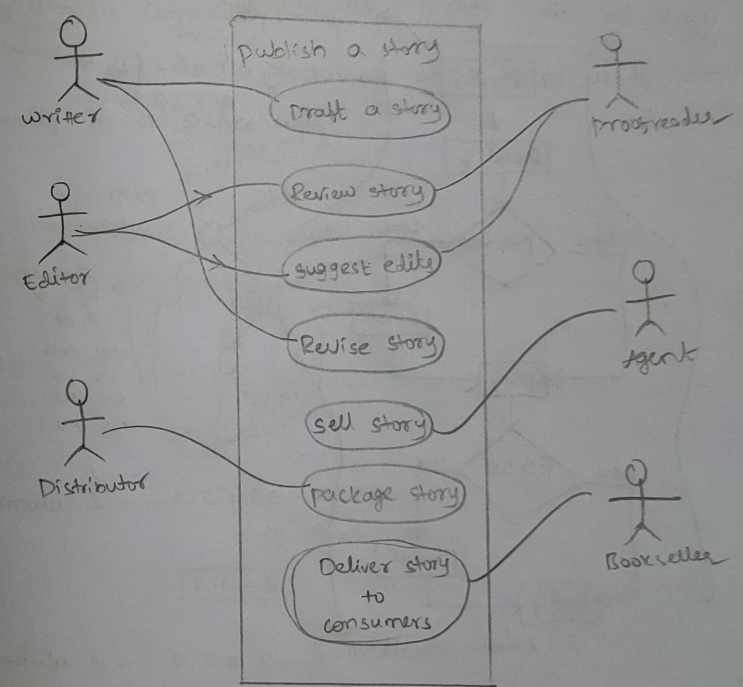
It is the functional black box testing technique that helps us to identify the scenarios that whole system can be tested.

- Here we have a diagram representing in the form of ACTORS and ACTIONS.
- Each Actor can have different Actions.
- Actor is a User and Actions can be modules where the access is given.
- characteristics of use case testing

Use Case can capture interaction b/w actor and system.

- Actors represent user and interaction is that each user take part into.

Ex:-



→ In above diagram 6 actors & 7 actions are there.

→ writter can do only 2-actions, editor also 2, protreader also 2 remaining all are doing can do only 1-action.

B) White box testing techniques:-

Example:- Min. Tests Required for statement coverage & Branch coverage :-

```
Read p  Read q
If p+q > 100 then
print "Large"
End if
If p>50 then
print "plarge"
End if.
```

options

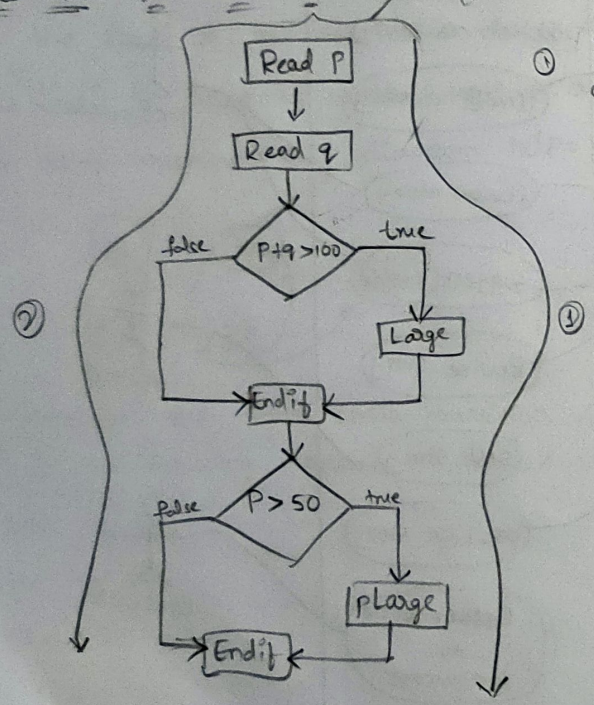
a) Statement coverage 2, Branch coverage 2

b) " " 1, " " 2

c) " " 3, " " 2

d) " " 4, " " "

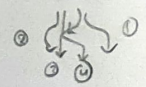
→ Flow chart for above pgm



branch coverage → P = 50, q = 50 ②

① P = 51, q = 50 } → statement coverage

Path coverage



Yes	Yes
No	No
No	Yes
Yes	No

→ 100% PC = 100% BC = 100% SC – true

100% SC not = 100% BC != 100% PC – true

100% SC = 100% BC = 100% PC – false.

→ statement Coverage, Branch coverage, Path coverage are the techniques which are available to cover all the statements, Branches & paths in a pgm.

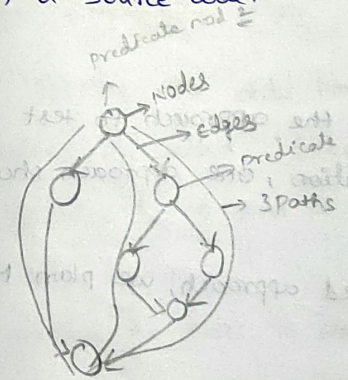
– This is usually used by developers while doing white box testing.

– It is also called as types of white box testing.

→ Cyclomatic Complexity:-

It is a technique which helps us to identify the Complexity in a source code.

Ex:-



Formula 1 – C.C = (E – N) + 2

= (8 – 7) + 2

$\boxed{C.C = 3}$

Formula 2 – C.C ⇒ no. of predicate nodes + 1

= 2 + 1

= 3

→ A Node which is having 2 (or) more edges (outgoing) i.e., predicate node.

Formula 3 – no. of regions + 1

= 2 + 1

= 3

Formula 4 :- C.C = no. of. paths = 3

→ To reduce the complexity of testing, source code should not be complex.

— This will be decided by manager & if needed source code has to be modified, if it is too complex.

C.C → 0 – 5 – Less complexity

6 – 10 – Medium complexity

>10 – high complexity

* Differences B/w Test Strategy & Test Plan :-

→ Test Strategy :-

Test strategy is the approach to test the s/w application. Based on the situation, one approach should be selected.

→ Test plan :-

Based on the selected approach, we plan how to take it forward.

* Difference b/w Functional & Non Functional Testing :-

Functional Testing	Non-Functional Testing
1) Functional testing Integration " System "	1) Performance Testing Compatibility " Globalization " Security " Accessibility " Usability " Reliability "
2) Checking for single users & single platform and single language – Focus on Normal people.	2) Checking for multiple users – Multiple platforms – Multiple languages.

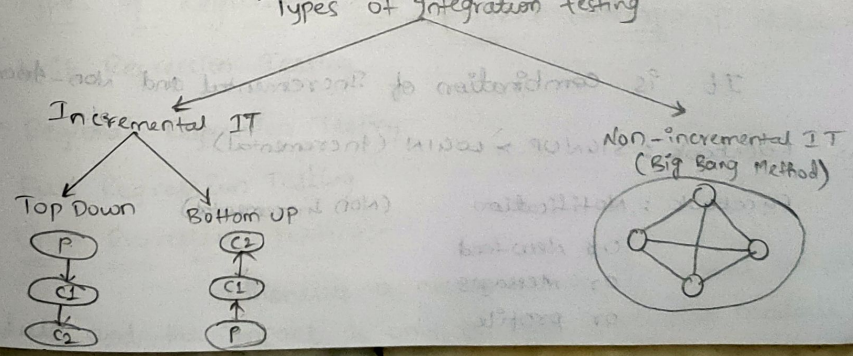
* Integration Testing :-

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

Ex:- In gmail, if we send a mail, the sent mail should be displayed in sent items.

– Checking whether it there in sent items or not is a integration testing.

Types of Integration testing



1) Incremental Integration Testing:-

TOP DOWN:- Whenever parent module is there, child module is added and we do Integration testing.

BOTTOM UP:- Whenever child module is there, Parent module is added and we do integration testing b/w these modules.

Examples:-

ATM: Insert card > pin > language > account type > enter amount > withdraw.

Flipkart: search > cart > buy > payment

Gmail: signup > Login

2) Non-Incremental Integration Testing:-

Whenever we cannot find which is a child and which is a parent module, even then we do integration testing between them, this is called "Non-Incremental integration testing". (BIG BANG).

Ex:- GMAIL: Compose
Inbox
Settings
Messages.

3) Sandwich Integration testing:-

It is combination of Incremental and Non-Incremental

Ex:- Facebook : SIGNUP > LOGIN (Incremental)

Facebook : Notification (Non Incremental)
or Newfeed
or Messages
or profile

→ STUBS & DRIVERS (Both are dummy modules):-

- STUBS:- Parent module is there
But child is not there

Then I will use stubs as dummy child module.

- Acts like a child module when child module is not present.

- DRIVERS:- Parent module is NOT there
But child is there

Then I will use drivers as dummy parent module.

- Acts like a parent module when Parent module is not present.

- Developers will create stubs and drivers when Real module takes lot of time to develop.

Ex:- For complex Applications
Hospital, Airways, Navy, submarine, space apps takes lot of time to develop.

- Dummy module will be used only for Testing purpose. It is not given to public or customer.

- Once real module is ready, stubs and drivers will be removed and Real module will be given for testing.

* Types of Regression Testing:-

1) Unit Regression Testing
2) Regional Regression Testing
3) Full Regression Testing

1) Unit Regression Testing:-

Whenever a new feature is added for existing module and the impact is only for that particular module, we

can also call it as unit.

→ When we do regression testing for that unit, we do not do regression testing for any other unit that is called "Unit regression testing".

2) Regional Regression testing:-

Whenever there is a change in one module and that module is having dependency with other few modules not all the modules, then we do regression testing for all the depen-dent modules, we call them as "regional regression testing".

3) Full regression testing:-

Whenever there is any change in the application and because of that change if we have an impact on the whole application then we do complete testing again for whole applicati-on. This is called "Full regression testing".

Ex:- 1) Hotse-App

Hall, kitchen, bedroom, bathroom, Pooja room

Ex:-2 Gmail

Login, compose, Inbox, sent, draft

Ex:-3 Flipkart

Electronics - discount changed - UNIT

Electronics, groceries - new offer - Regional

All products - new offer - Full

Note:- Usually in Initial builds (not first build), we perform UNIT Regression Testing.

→ In the middle of the project, we perform Regional regression testing.

→ At the end of the project, before giving application to customer, we perform "Full regression testing".

* Types of RTM:-

Ex:-

Business Req#	Test Scenario#	Test case#	Defects #
BR1	TS1	TS1, TC1 TS1, TC2	DO1
BR2	TS2	TS2, TC1 TS2, TC2 TS2, TC3	DO2 DO3
BR3	TS3	TS1, TC1 TS2, TC1 TS3, TC1 TS3, TC2	NIL

1) Forward Traceability Matrix:-

Mapping from BASE document to DERIVED docume-nt.

Ex:- Req > scenario > Test case.

2) Backward Traceability Matrix:-

Mapping from DERIVED doc. to BASE document.

Ex:- Test case > scenario > Req.

3) Bi-directional Traceability Matrix:-

Mapping from DERIVED document to BASE document and Mapping from DERIVED Document to Another DERIVED document.

Ex:- Scenario > Req.
Scenario < testcase

* Defect Seeding:-

Injecting the defect in the application by developer to check the efficiency of test engineer is called "defect seeding".

- This is done based on the req. of manager.
- If Test engineer is not working properly or if manager get a doubt on test engineer performance, then defect seeding will be done secretly.

* Defect Masking:-

Whenever one defect is hiding another defect that is defect masking.

Ex:- TE - Gets Build 01

Login - A - Defect raised (Masking defect)

Compose B - Login defect is hiding compose defect in build 01. (Masked defect)

Build 02

Login A - Retesting

Compose B - Defect

* Defect leakage:-

The defect which is not found by the testing team but it is found by the customer or end users it is called defect leakage.

- For a good testing defect leakage percentage should be 0%

* Defect Triage Meeting / call:-

Manager arranges a meeting with all the development team and testing team and discuss about the existing defects which are not fixed because the defect age should not be increased. And defect status should be moved for further level until the defect status is closed.

- This meeting is known as defect triage meeting. It is the main part of agile meeting.

→ Apart from closed status, all are existing defects.

(new, assign, fixed, reopen, can't reproduce, can't fix, deferred, invalid, rfe, duplicate)

Main intention of this is to close the defects as early as possible.

→ If defect is raised on 10th may and today is 19th may, AGE of this defect is 9 days.

* Defect density:-

Defect density = No. of defects (valid) / Total size of the project (LOC / Duration)

Line of code.

Ex: 1

DD = 80 / 5000 (LOC)

DD = 80 / 100 (days)

Ex: 2 DD = 100 / 2000 (LOC)

DD = 100 / 6 months (duration)

→ No. of. defects means the defect which are considered as valid defect not all the reported defect.

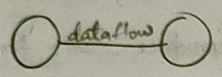
* Latent Defect:-

It is a defect which is found once application is released to production server and that defect is there in production server for long time.

TE → production server - 2, 3, 4 yrs te, cust, end user

* SIT (System Integration Testing):-

Testing the data flow b/w 2 or more systems.



Ex:- fB — Gmail

* Principles of Software Testing:-

→ There are 7 principles

1) Exhaustive testing is not possible:-

Practically we can't do exhaustive testing since it contains too many scenarios.

- So we go for optimized testing.

2) Defect clustering:-

There is a person saying that 80% defects will be found in 20% module. This is from Pareto principle.

Ex:- If there is a defect in login module, surrounding login module will have more no. of defects. This login area can be called as 'Defect cluster area'.

3) Pesticide paradox:-

Same kind of testing or scenarios will not help us in finding new defects.

→ Always we need to update our testing type and scenario to find new defects.

Ex:- If we want to get rid of mosquito, we try with different solutions like:

All out / Good night or Hit or Net or bat

4) Testing shows a presence of defects:-

When we do testing, we can find defects.

5) Absence of Error - fallacy:-

Java - req 1
.net - req 2

It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement.

→ This can be avoided by doing static testing.

6) Early Testing:-

Perform the testing from initial stage of SDLC (ex: review).

→ That is static testing. If we do this, we can avoid the wrong things which can happen.

7) Testing is content dependent:-

For different domain, our testing should be in diff. way.

→ Similar type of testing for all applications is a bad practice.

→ we have to perform based on the type of application.

Ex:- If Banking, my thought is more on Security.

If gaming, my thought is more on Adhoc.

* TEST PLAN:-

Test plan is a document which is prepared for future testing activities.

- It is usually prepared by the test lead or manager.
- Test engineer can also involve - if there is any support needed for test lead or manager.
- Test plan contain different sections or attributes like:

1) Objective:- This section tells about the purpose of preparing the test plan.

2) Scope:- This will say the limitations.

This section will tell about

a) Features what to be tested. (Ex: Gmail)

b) " " not to be tested (Ex: Fb, insta)

3) Schedule and milestone:-

This section will tell which activity has to done first and which activity has to done next.

- It is just like a time table of the project.

Ex:- Req - wtc - Etc - AT (milestone) - PR (milestone)

Ex:- Req (1 april) > write test cases (20 apr) > Execute test cases (10 may) > UAT (30th may) > GO LIVE (5 June)

4) Entry and exit Criteria:- This section tells about when to enter and when to exit each type of testing.

Ex:-1 Entry criteria for Functional Testing:

WBT should be done

Build has to be installed to Testing server.

Exit Criteria

Pending defects should be lesser than or equal to:

0 Blocker
1 critical
10 Major
20 Minor

Ex:2 Entry criteria for IT:

0 Blocker
1 critical
10 Major
20 Minor

Exit criteria for IT:

Pending defects should be lesser than or equal to:

0 Blocker
0 critical
5 major
10 minor

Ex:3 Entry Criteria for ST:

0 Blocker
0 critical
5 major
10 minor

Exit Criteria for ST:

Pending defects should be less or equal to

0 Blocker
0 critical
0 major
5 minor

5) Defect tracking:- This section will tell about whenever a defect is found, how to track the defect and which is the tool we are using to track the defect. And what are the terminologies we are using to raise the defect.

Ex:- ALM tool

P1 or P2 or P3 priority

Blocker, critical, major, Trivial Severity

open, confirm, In progress, Resolved, verify Status

6) Assumptions:-

This section will tell about what are the assumptions we have during this project.

Ex:- All employees will be there in the office daily.

7) Risk:- This section will tell about what are the possible risk happen during the project.

Ex:- All employees are not in office. Leave/Quit.

8) Contingency plan or mitigation plan or Backup plan:-

This section will tell about how to over come the risk which occurs during the process.

Ex:- Arrange regular KT among the employees when every-one are there in office, before Risk occurs.

9) Roles and Responsibilities:-

This section will contain what are the roles and responsibilities for team members.

a) Ex: Roles and responsibilities for test engineers (Manual):-

Going through the requirement and understanding the requirement and identifying the scenarios and writing the test cases, prepare RTM, and execute test cases when the build is given, find the defect and raise the defects etc.

b) For automation test engineers

we should perform above task plus we should write test scripts and execute test scripts by using the selenium tool. Manage test Script.

10) Environment/platform:- This section contains which platform we are using for testing purpose.

Ex:- windows 10, 8, 7, chrome browser.

11) Deliverables:- This tell about what are the documents that should be prepared for the project.

Ex:- Test plan, Test cases, RTM, Defect reports etc.

12) Graphs and metrices:- This section contains the graphs and metrices that are prepared for project.

Ex:- Bar Graphs, pie charts, ... etc.

* Acceptance testing:-

It is a testing which is mainly done to check the business scenarios of the application which is done by the customer.

(or)

It is an end to end testing which is done by the customer to ensure the application is fit for business or not.

Ex:- For courier business - placing a courier, shipping, tracking, delivery date, weight, amount, agent, status, return option, otp, address is tested.

- But cancel scenario is forgotten by TE. This is tested by customer and he may find defect.
- If customer is checking in company, it is alpha.
- If he is checking in his place, It is Beta.

Why acceptance testing is done?

1) This is mainly done to get a confidence for customer before he releases the product to the end users.

2) Because of business pressures, software company might be releasing the application to the customer with some defects. To ensure this is not there, customer will do acceptance testing.

3) If the product is released to the end user without checking the business scenarios, it will affect the customer business. To avoid this, acceptance testing has to be done.

4) We may forget some of the business scenarios to test and those scenarios would be tested by the customer.

Note:- If the customer finds the defect during the acceptance testing. That is bad for test engineers.

So before giving application to customer, we should think all those business scenarios and find defects.

→ **Types of Acceptance Testing:-**

Alpha & Beta testing are types of acceptance testing.

Difference b/w alpha & Beta testing:-

Alpha testing	Beta testing
1) Alpha testing is performed by testers who are usually an internal employee of an organization.	1) Beta testing is performed by the client or end users who are not an employee of an org.
2) It is performed at the develo-per site (s/w company).	2) It is performed at the client location or end user of product.
3) It involves both black box and white box testing techniques.	3) It typically uses black box testing techniques.
4) It requires lab environment or testing environment.	4) It needs real time environment
5) Virtual environment.	5) Real time environment (production)
6) Done at offshore (company)	6) Done at onsite (customer place)
7) It is done under controlled environment.	7) It is done under uncontrol-ed environment.
8) It is closed for public.	8) It is open for public.

- Hyd (company) - dev and test ------ offshore
- Client (Pune) - customer place ---- onsite

→ Approaches of acceptance testing. or who can do AT?

Approach no 1: BA or IT engineers of customer will do acceptance testing at customer place - BETA

Approach no 2: Employees or test engineer of customer will do acceptance testing at customer place. - BETA

Approach no 3: Test engineers of s/w company will do acceptance testing at customer place - BETA

Approach no 4: Test engineer or BA of software company will do acceptance testing under the control of customer at developer place. - ALPHA.

* Hot fix/incident management System:-

When the software company releases the product to customer/end user, if the end user/customer finds any defects in the software, customer will raise an incident to software company and software company should fix the defect as early as possible and give back to the customer. This situation is called 'hot fix'.

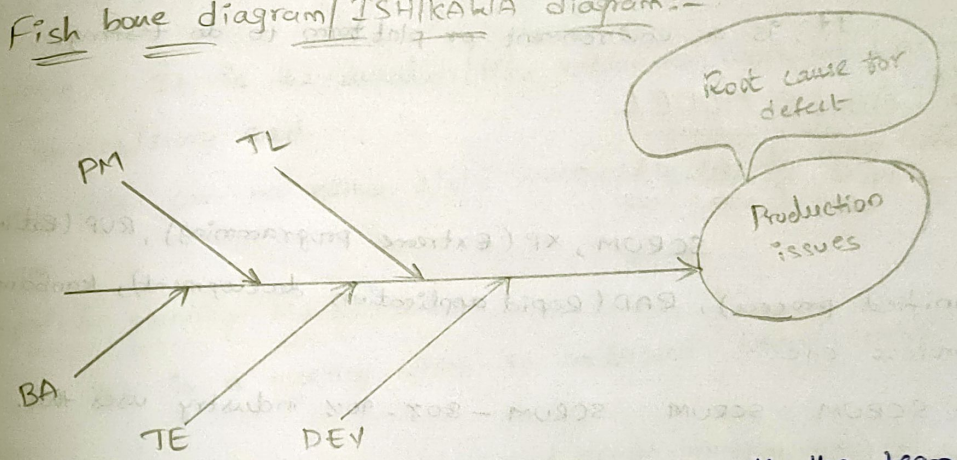
If any problem is find by the customer or end user in the production sever, it will be raised as the incident to the software company. Every incident is raised in the form of ticket and every ticket will have a priority.

The commitment b/w software company and customer is called Service level agreement (SLA).

Ex:-

Priority of Ticket		Time give to fix
P1	-	3 hrs
P2	-	10 hrs
P3	-	1 day
P4	-	3 days
P5	-	10 days.

* Fish bone diagram/ISHIKAWA diagram:-



Whenever there is a production issue, all the team member will gather together and they discuss about reason for production issues that is documented in the form of fish bone diagram. The main purpose of this is to find the root causes of the production issue.

→ It is done after Hot fix, otherwise it is not done.

→ It is mainly for future purpose, because of this Employees will work more effectively and production defect (issues) can be avoided.

* Short Term Release / Interim Release :-

Between two planned releases of an application. Some times customer gives a unplanned release which is done in short duration. It is called short term release.

ex:- AAA - planned - apr 1

KAA - unplanned - apr 15 - short term release

BAA - planned - may 1

→ What is Test bed?

It is a environment or platform to do testing.

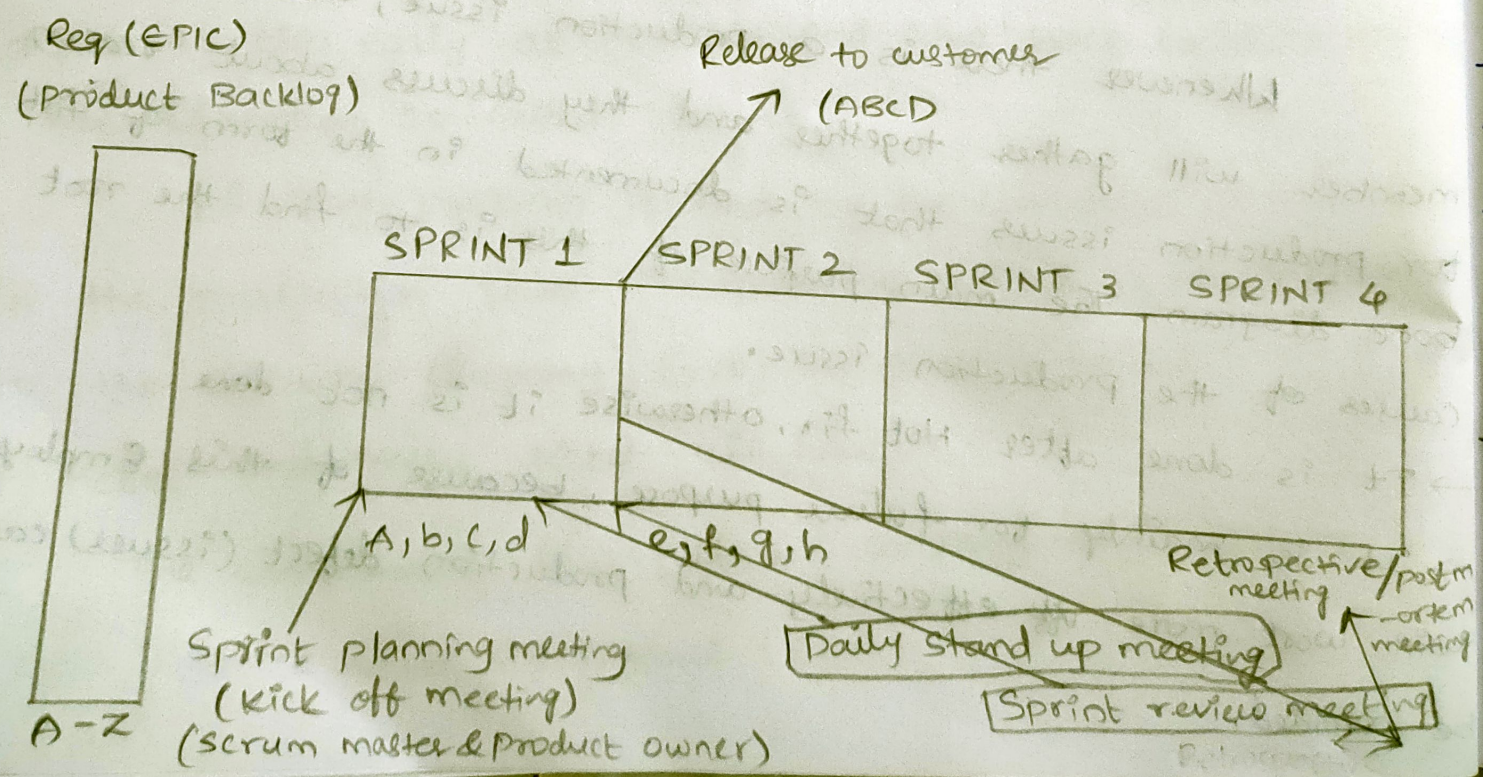
*** AGILE MODEL

→ Types of Variants:-

SCRUM, XP (Extreme programming), RUP (Rational unified process), RAD (Rapid application development), kanban process etc.

- SCRUM SCRUM SCRUM - 80% - 90% industry uses this.

AGILE (SCRUM)



→ EPIC :- It is complete set of Req given by customer

A to Z

- Epic is also called Product Backlog.

- User Story / Story Card :-

It is complete a part of the Req.

A - User story

B - User story

C - User story

→ Sprint :- It is the duration / time taken to work on 1 or more user stories / story card.

- Each sprint can be either 2/3/4 weeks. It depends upon customer Decision.

→ Sprint planning Meeting :-

It is a meeting which is conducted before sprint starts.

- In this meeting product owner and scrum master will be involved.

- They will decide which user story has to be worked on that particular sprint. Also, they will decide the duration of the sprint.

→ Who is Product owner?

He is a person who is a customer or representative of a customer.

- He decides which are the important user stories.

→ Who is Scrum Master?

He is a person who is representing software company. Scrum master can be Project Manager / Dev lead / Test lead depend on project decision.

→ Daily stand up meeting :-

All team members will gather together like PM, dev lead, test lead, test engineers, developers, BA. Customer can also involve in this meeting.

Here, each employee should give work update every day.

- If the meeting is in the morning, what was done yesterday and what will be done today, should be updated.
- If the meeting is in the evening, what is done today and what will be done tomorrow should be updated.
- This is very much helpful for a better communication between team members.

→ Sprint Review meeting :-

In this meeting, scrum master will check whether all the user stories are completely developed and tested and is it ready to release to customer or not.

- This happens at the end of the sprint.

→ Sprint Backlog :-

It is a user story which is not completed in that particular sprint. That is carried forward to next sprint.

Ex :- If 'D' user story is not completed in sprint 1, it is a backlog which is taken from sprint 1 to sprint 2.

→ Product Backlog:- It is the user stories which is taken for a new sprint.

Ex:- e, f, g, h are from product backlog for sprint 2.

- In the same way, till all the user stories are completed, multiple sprints will be going on and full application will be ready.

→ Note or advantages:-

1) From this model, work is getting completed very fast.
2) It is the most successive model currently.
3) we can work more within less time.
4) communication is very good b/w company and customer.
5) we can handle any pressure.
6) It is best model for Existing projects.

→ How to Explain about project with respect to model, build, release?

Chats ------ V and V ------ release 1 (6 months) ---- 30 builds.

Status ---- agile model ----- release 2 (4 weeks) ---- sprint 1 ---- 6 builds

Calls ---- agile model ---- release 3 (4 weeks) ---- sprint 2 ---- 5 builds.

Web ---- agile model ---- release 3 (4 weeks) ---- sprint 3 --- 4 builds

Video call ---- agile model --- release 4 (2 weeks) ---- Sprint 4 --- 3 builds

Payments ---- agile model ------ release 5 (2 weeks) --- Sprint 5 --- 7 builds