MANUAL TESTING

SDLC: - software development life cycle

- It is standard procedure to develop new software

It has different stages like

REUIREMENT

FEASIBILITY STUDY (OR) ANALYSIS

DESIGN

CODING

TESTING

INSTALLATION

MAINTANANCE

It has different model

- Waterfall model
- Spiral model
- V. model
- Prototype model
- Derived model
- Hybrid model
- Agile model

Waterfall model: -

It is step by step procedure (or) A standard procedure to develop a new software.

REQUIRMENT COLLECTION: -

It is nothing but collection of requirements from customer's place. it is done by business analyst or product analyst where in BA will go to customers place and collect the requirement in

the business language and convert into software language and explain the requirement to developer, test engineer, project manager, architect etc. this process is called requirement collection.

Here BA acts like bridge between customers and IT companies.

WHO CAN BECOME BA?

Domain expert: -

A person who has worked on the same domain for 10 to 15 years and has got very good knowledge can be called as domain expert

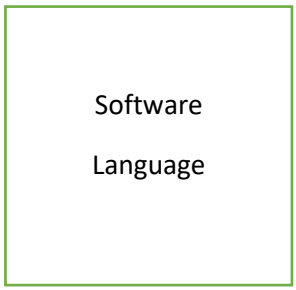
=>**Senior development engineer /senior test engineer:**

Who has got more than 6 to 8 years of experience on the same project and has got very good project knowledge can become BA.

NOTE: -

- For 70% to 80% of critical, complex and longform project BA will be there.
- For 20% to 30% of simple and small projects senior developer or senior test engineer can play the role of BA.

Customer /clint =>BA=>IT company



CRS (customer requirement Specification)

SRS (software requirement specification)

BRD (business requirement documentary)

FRD (functional requirement documentary)

BS (business specification) **FS** (functional specification)

User stories

Feasibility study:

Once after the requirement collection is completed then only, we go for feasibility study. This is done by team which consist of project manager, BA, Architect, Finance team and HR team.

►This is the stage where company will decide whether to take up the project or not and if we take up the project company will check do we have sufficient resources, sufficient technology and sufficient lab set ups.

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

Roles of PM: Project manager will interact with Architect, BA, Finance officer and HR team and gather the information and then PM will decide to take up the project or not.

► **Roles of Architect:** Architect will think from technical point of view and he will decide technology is it possible to implement the project or not, if yes which technology should be followed all these decisions will be taken by Architect

►**Role of Finance team:** Finance team will think from money point of view, they will see how much should be invested and what is the operational cost and we invest money do we get profit or not.

►**Role of HR:** HR team will think from resource point of view. they will think how many experience engineers and freshers should be hired to work on the project

Design:

‣ Once after the feasibility study is completed then we go for design phase. It is done by architect.

► There are 2 types of design

1. High level design (HLD).

2. Low level design (LLD).

HLD: The design of Architecture of project. It is done by Senior Architect.

LLD: The detailed design of smallest feature is called as LLD. It's done by senior developer.

Coding:

*It is nothing but writing the programmes. It is done by development engineers where in they implement the software according to the Specification, Customer Requirement.

Testing:

once after coding is completed the next phase is testing. It is done by test engineers, here testers will start testing the application according to the customers requirement specification in order to find the defects in the software is called as Testing.

Installation:

Once after the testing is completed and the software is ready, we will install the software in the customers place so that customers can start to use the software and run the business.

There are 2 types of installation.

1. Single time installation

2. Multiple installation

Note:

Single time installation can be done by Senior Development egg.

Multiple installation is done by field engineer / installation engineer /implementation engineer / support engineer.

Maintenance:

Once after the Software is installed Customer will the start to use the Software and own run the business, while using the software. its customers face any problem or issue then Company will fix the problem or defects for free and reset the defect and give new Software to the Customer freely for a period of 6 months to 1 year depending on the agreement between the customer and the company. this phase is called as maintenance.

Waterfall model: It is step by step procedure (or) A standard procedure to develop a new software.

WHE THIS MODEL IS CALLED WATERFALL MODEL?

Here the backtracking is not possible i.e., once after the feasibility study is completed all the requirement will be freeze and we cannot go back and change the requirement so this model is called as waterfall model.

DRAWBACK OF WATERFALL MODEL:

1. Cannot adopt the changes in requirements

2. Testing is a small phase which is done only after coding.

3. Rework cost is high

4. For bigger and complex projects, this model is not good as a risk factor is higher.

5. Not suitable for the projects where requirements are changed frequently.

ADVANTAGES OF WATERFALL MODEL.

1. Simple and easy to understand and use.

2. it is easy to maintain.

3. Initial investment is less.

4. Since requirements are not going to change, we can expect very good quality of the product.

APPLICATION OF WATERFALL MODEL:

1. To build small application.

2. To build short term application.

3. Whenever we are sure that requirements are not going to change.

WHAT ARE THE DISADVANTAGES OF DEVELOPER TESTING THE APPLICATION.

1. Developers will utilize the time allocated for testing to build the application.

2. Developers will be over confident.

3. They will concentrate more on development rather than testing.

4. They will not test the software from negative point of view.

5. Even though bugs are there in the software they may hide it.

6. Lack of End-to-End knowledge.

SPIRAL MODEL:

Spiral model is a step-by-step procedure or standard procedure to develop the new software. In order to overcome the drawback of waterfall model we go for spiral model. spiral model is mainly used when there is dependency between the modules or features or functionalities.

ADVANTAGES:

►Requirement changes can be done after every cycle.

►Customers get an opportunity to see the software in every cycles.

►Testing is done in every cycle before going to next cycle.

►Spiral model is a controlled model. Ime When one module is completely stable then only, we develop next module.

Drawbacks:

►Requirement collection and design phase are not tested

Is not beneficial for smaller projects.

►Spiral may go infinitely.

►Documentation is more as it has intermediate phases.

►It is costly for smaller projects.

►**Why spiral model is called iterative model?**

In every cycle same process is repeated so the model is called iterative model.

►**Why this model is called Incremental model?**

In every cycle we keep on adding new module, so this model is called incremental model.

V-MODEL (VERIFICATION MODEL OR VALIDATION MODEL)

It is step by step procedure or standard procedure to develop a new software in order overcome the drawback of waterfall model and spiral model we use v-model here all stages are tested

Verification:

Verification CRS, SRS.HLD, LLD again the requirement and check whether if it is matching the requirement or not is called verification it is done by the testers before the software is develop

Here we ensure that are we building software right, system right product right or application right.

Validation:

Verifying the functionality of an application by executing test cases is called validation it is done by the testers once after the software is develop.

Here we ensure that are we building right product.

Advantages:

- Total time is less (testing is done from early stage)
- All the stages are tested
- Since testing is done in the early-stage downward flow of defect will be less.
- Total investment is less (downward flow of defect will be less and cost of fixing the defects is less)
- Software quality will be good.
- Requirement changes can be done in any stage

Drawback:

- Initial investment is high.
- Documentation is more.

Application:

- The requirement is well defined and not ambiguous.
- Project medium size is to large size.

- When costumers want to quality software which is in short span of time.

Prototype model:

It is step by step procedure or standard procedure to develop a new software

- Whenever the customer has no clarity about the requirement, we will go for prototype model.
- Here we create a dummy software and show to the customer
- Until the customer like the dummy software, we will go for agreement after successfully completed agreement we go for actual software.

Advantages:

►Customer will get to know how the software looks in the early stage itself.

►There will be improved communication between customers and developers.

► Requirement changes are allowed.

►Missing functionalities can be easily figured out.

► Errors can be detected much earlier thereby saving a lot of effort and cost.

►The developed prototype can be reused by the developer for other projects in the future.

DRAWBACK :

- Total time take is high
- Total investment is more
- There will be delay in releasing software to customer

- Poor documentation due to continuously changing customer requirements

APPLICATIONS:

- When the customer are not having the requirement .
- When ever the customer is new to the software .
- When the developers are new to the domain .
- When the customer gives the requirement in stages .

DERIVED MODEL :

Here we take a basic model or any single model and any single model and change it according to the project needs to company standards is called derived model.

HYBRID MODEL

The process of combining or merging more than one model into single model is called hybrid model.

EX:1

Combination of prototype model

- Where the module are dependent and when the customer not having clarity about requirement
- Not having clarity requirement prototype model

EX:2

We combine prototype +V.model➔customer ,not having clarity about project and they wants quality of product in tight shedual.

SOFTWARE TESTING

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

Or

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

Or

Executing of programs with the intent to find the defect in software is called as software testing.

TYPES OF SOFTWARE TESTING :

1. White box testing
2. Black box testing
3. Grey box testing

WHITE BOX TESTING :

- Testing each and every line of the program is called white box testing.
- It's done by the developer in order to reduce the bugs, before given the software to testing team
- WBT is also called as
  - Unit testing
  - Clear box testing
  - glass box testing
  - Transparent box testing
  - Open box testing
  - Code base testing

BLACK BOX TESTING

- Verifying the functionality of an application according to the customer requirement specification is called BBT.
- It is done by the test engineer in order to find the defects.
  - Functional Testing
  - Specification based testing

- closed box testing
- Behavioural testing

Difference between WHITE BOX TESTING AND BLACK BOX TESTING:

SL.no	WHITE BOX TESTING	BLACK BOX TESTING
1	Checking the each and every line of the program	Verifying the functionality of an application
2	Its done by developer	Its done by testers .
3	We need to know programming	We need not know programming
4	We need to know the internal design of source code	We need not to know the internal design of source code
5	source code is visible	source code is not visible

TYPES OF BLACK B0X TESTING :

- Functional testing
- Integration testing
- System testing
- Acceptance testing
- Smoke testing
- Adhoc testing
- Exploratory testing
- Usability testing
- Performance testing
- Regression testing
- Compatibility testing
- Reliability testing
- Recovery testing
- Globalization testing

What are the levels of testing :

- Functional testing
- Integration testing
- System testing
- Acceptance testing

GRAYBOX TESTING :

The combination of white box testing and black box testing is called graybox testing.

WHY WE DO SOFTWARE TESTING :

- Go gain customer confidence .
- In order to provide good quality of the software to the customer.
- To identify defects in the software.
- To avoid extra costs.
- To avoid risks .

USABILITY TESTING :

- Testing the user friendliness of an application is called as usability testing
- No special training should be required in order to use the application.
- The effort spends on using the application should be less.
- The no.of clicks/action performed on application should be less.

WHO SHOULD DO USABILITY TESTING:

It is always a good approach if the end users test the usability testing because by the time we perform usability the test engineer might have perform. All the types of black box testing so that it becomes user friendly to the test engineer.

ON WHAT BASIC WE WILL PERFORM USABILITY TESTING:

- All the frequently use features should be in top to bottom approach or left to right approach.
- The end user will judge the application based on look and feel of the application
- The application should be according to the GUI standards and it should be pleasant to the eyes and attractive.

WHEN TO DO USABILITY TESTING:

- When the customer is very particular about the user friendliness of an application we will perform usability testing from initial built it self.
- If the customer is not particular about the user friendliness after performing all the types of black box testing . we at the end to do usability testing.

WHAT APPLICATION ON WE PERFORM USABILITY TESTING:

- On multiple users access application
- On more revenue generation application
- On banking application (phonpe,gapy,paytm etc)
- The application which will be expected to be very simple (ex:games)

ACCESSABILITY TESTING(ADA TESTING)

Testing the functionality of an application whether it is accessible for physically challenge or physically disabled people is called as accessibility testing.

Some of the examples are:

1. Voice recognition
2. Speech to text
3. Camera gestures
4. Subtility
5. Zoom in zoom out
6. Turn by turn navigation

HOW TO DO ACCESSABILITY TESTING:

Normal test engineer cannot perform accessibility testing so we depend on automation tools.

Some of tools are:-

1) Wave web accessibility
2) A checker
3) A viewers
4) In focus

The condition which are verified in accessibility testing are colour , font, size and style of the application should be according to GUI standards

ADA: American disability act testing

FUNCTIONAL TESTING:

Testing each and every component of an application Thorley or rigorously (deep testing) again the CRS is called as functional testing .

It is also called as component testing or field level testing.

FIRST NAME TEXT FIELD:

Application: Gmail

Module name: login /signup page

CRS: 4 to 16 characters accepted

@accpeted

First letter should be capital

Alpha numerical accepted

No blanks ,no spaces

‘_’ accepted

POSITIVE SCENARIOS:

1) It should accept minimum 4 character.
2) It should accept maximum 16 character.
3) It should accept @ symbol only once.
4) It should combination of numbers and alphabets.
5) It should accept only first letter is capital
6) Is should accept only _ symbol
7) It should accept only upper case alphabets also.
8) It should accept only lower case alphabets also.
9) It should accept combination of upper and lower alphabets.

NEGATIVE SCENARIOS:

1) It should not accept below 4 character.
2) It should not accept above 16 character.
3) It should not accept only numbers.
4) It should not accept only first letter is lower case.
5) It should not accept only lower case of alphabets.
6) It should not accept @ symbol more the once.
7) It should not accept blanks.
8) It should not accept special characters apart from @ character.
9) It should not accept spaces .
10) It should not accept emoji’s

POINTS TO REMENBER WHILE DOING FUNCTIONAL TESTING:

i. We must always start testing the application with the valid data.
ii. If the application is working for valid data only then we must test for invalid data.
iii. If the application is not working for one of invalid values, we can continue testing for the other invalid values.

➔In testing ,we should not assume or prepare requirement ,if we have any queries, talk to the one who knows the requirement very well and clarify the queries.

➔we must not do ever testing (testing for all possible junk values) or under testing (testing for a set of values) we must only try to do optimes testing.

➔we must do both positive testing(testing of valid data) and negative testing (testing for invalid data)

TYPES OF TESTING:

Any testing we can do it in 3ways:

1. Over testing
2. Under testing
3. Optimized testing

OVER TESTING:

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

UNDER TESTING:

Testing t the application with the insufficient set of data is called as under testing by doing under testing we miss lots of defects.

OPTIMIZED TESTING:

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

ENTRY:

1. White box testing should be done.
2. The software should be installed in proper test environment.

3. Test cases should be ready.
4. Test data should be ready.
5. Resources should be available.

EXIT:

It is based on the following aspects

1. 85% to 90% of the test cases should be executed.
2. 85% to 90% of the test cases should be passed.
3. No critical & blocker bugs are allowed.

Steps to do functional testing:

- ⇨ Prepare functional test plane.
- ⇨ Prepare function test scenarios & test cases.
- ⇨ Execute test cases.
- ⇨ Report the defects.
- ⇨ Track and retest the defects.
- ⇨ Retesting & testing goes on until functional testing is completed.

TEST CASE:

Test case template(Header)

Test case name/test case ID:

Project name:

Release name:

Requirement ID:

Module name:

Pre condition:

Test data:

Priority:

Test case type:

Brief description:

BODY OF THE TEST CASE:

Step.no	Action/description	input	Expected result	Actual result	Status

FOOTER:

AUTHOR NAME:

CREATED DATA:

REVIEWED :

APPROVED BY:

60.0 amount transfer (SBI)

60.1 from accept 12-digit integer

60.1.1 Should accept only those account which are created by manager

60.2 to account number test field.

60.2.1 should accept 12 digit integer.

60.2.2 should accept only those accounts which are created by manager.

60.3 amount text field.

60.3.1 should accept only positive integer in the range 100 to 50000.

60.3.2 should not accept more than balance.

POSITIVE SCENARIOS:

⇨ It should accept account number 12 digits only.

⇨ It should accept account number text field only.
⇨ It should accept only the accounts which are create by manager.
⇨ It accept account number should be positive integer only.
⇨ It should accept range of amount 100 to 50000 only.

NEGATIVE SCENARIOS:

⇨ It should not accept more than 12 digit account number.
⇨ it should not accept account number any other field.
⇨ It should not accept the account which are not created by manager.
⇨ It should not accept negative integers in account number.
⇨ It should not accept account number without numbers.
⇨ It should not accept less than the amount of 100.
⇨ It should not accept more then the amount of 50000.

Test case template (header):

Test case name (test case ID:SBI-AMOUNT-TRANSFER-R01)

Project name:SBI

Release date:R01

Requirement ID: CRS -60.0

Model name: amount transfer

Pre condition : test URL, user name & password, account numbers ,basic details

Test data: s.purushotham,**********,www.SBI.com,9989881194,8985895484.

Priority :

Test case type: functional testing case.

Brief description : to validate that the amount transfer feature of SBI BANKING application .

BODY OF THE TEST CASE:

Step.no	Action/description	input	Expected result	Actual result	Status
1	Launch a browser and enter the test URL	www.SBI.com	Login page should be displayed.		
2	Login as user ‘A’ with valid credentials.	s. purushotham **********	Home page should be displayer.		
3	Click on amount transfer link	N/A	Amount transfer page should be displayed.		
4	Following are the step to test to account number component	N/A	N/A		
A	enter 12 digit valid account number	895623215487	Its should be accept		
B	enter less than 12 digit account number	789456232	It should not accept		
C	account number enter in text field only	N/A	It should accept		
D	Account number should be number only	897469846465	It should accept		
F	Accept only number which are crated by manager	N/A	It should accept		
G	Enter amount range of 100 to 50000	50000	It should accept		

H	Enter amount less than 100	99	It should not accept		
I	Enter amount greater than 50000	50001	It should not accept		
J	Enter greater than 12 digit account number	895623215789	It should not accept		
K	Enter account number outside the text field	N/A	It should not accept		
L	Enter negative account number	-8-9886868676	It should not accept		
M	Enter account number which are not created by manager.	N/A	It should not accept		

INTEGRATION TESTING:

Testing the data flow between dependent module is called as integration.

NOTE:

We will test the data flow only in one direction or we might test in both the directions . we might not test the data flow to all the modules because it varies from module to module.

APPLICATION:(google drive)

1. Login to google drive, click on (+) compose button and select the file and click on upload button and check weather it is display in.
2. Login to google drive ,application and delete ,the select file ,check weather the deleted file is displayed in bin.
3. Login to google drive click on bin select a file and restored.

PROCEDURE TO DO INTEGRATION :

I. First we should understand the complete project it mean we should understand how each & every feature works.
II. We should understand how the feature are dependent.
III. Identify all possible scenarios.
IV. Prioritise the scenarios.
V. Write integration test cases.
VI. Test the application by executing test cases.
VII. While executing if we find out any defects we have to report them to the development team.
VIII. Track and retest the defects.
IX. Tracking and retesting will continue until integration testing is completed.

TYPES OF INTEGRATION TESTING:

There is two types of integration testing:

1. Incremental integration testing.
2. Non incremental integration testing.

NON-INCREMENTAL INTEGRATION TESTING:

It is a type of integration testing where all the modules are integration at once and tested as a unit.

➔it is also called as big bang approach.

DISADVANDAGES :

- Identifying the root cause of the defect id difficult.
- We might miss some integration scenarios.
- Chances are there where it may prove to defect.
- We have to wait until the entire application is ready.
- We will have less time to test the application.

INCREMENTAL INTEGRATION TESTING:

Incrementally adding the modules and testing the data flow b/w the dependent modules is called as incremental integration testing.

There are two types of incremental testing

1. Top-down incremental integration testing
2. Bottom-up incremental integration testing

TOP-DOWN INCREMENTAL INTEGRATION TESTING:

Incrementally adding the modules and testing the data flow between the dependence modules but make sure that the module we have added is the child module is the previous module.

BOTTOM-UP INCREMENTAL INTEGRATION TESTING:

Incrementally adding the modules and testing the data flow between the dependent modules but make sure that the module we have adding is the parent module of the previous module.

TOP-DOWN INCREMENTAL INTEGRATION TESTING:

ADVANTAGES:

Fault localization (finding out) is easier.

DISADVANTAGES:

Needs many study.

BOTTOM-UP INCREMENTAL INTEGRATION TESTING:

ADVANTAGES:

Fault localization (finding out) is easier.

No time wasted waiting for all module to be developed unlike big bang approach.

DISADVANTAGES:

Critical modules which control the flow to application are tested last and may be prone to defects.

DIFFERENCE BETWEEN STUDYS AND DRIVERS:

STUBS	**DRIVERS**

Stubs are used in top down integration testing.	Drivers are used in bottom up integration testing.
Stubs are basically known as a "called program"	Drivers are the "calling program"
Stubs are basically used in the absence of low level modules	Drivers are mainly used in absence of high level modules.
Dummy program of lower level modules	Dummy program of high level modules

ENTRY:

1. White box testing should be done.
2. The software should be installed in proper test environment.
3. Test cases should be ready.
4. Test data should be ready.
5. Resources should be available.
6. It should have met with exist criteria of functional testing.

EXIT:

1. All test cases should be executed.
2. 90 to 95% of the test cases should be passed.
3. No critical & blocker defects should be there in the software.

SYSTEM TESTING:

It can also defined as, it is an end to end testing where testing environment should be similar to production environment /

Why it is end to end testing:

Testing navigate through all the feature and check whether the end feature or latest feature is working as expected or not.

Example for end-to-end testing:

CBO-CRS-1

If any new customer apply overdraft for the first time then the back should charge 2% rate of interest per month and also they should charge an activation fees of RS.250

If the same customer apply over draft for second time the bank should charge only 2% rate of interest per month and bank will not charge activation fee.

TYPES OF ENVIROMENT:

1. Development environment
2. Testing environment
3. Production environment

DEVELOPMENT ENVIROMENT:

In the development environmental develop the software .which is access to development team . in this development environment development lead and senior development engineers.

TESTING ENVIRONMENT:

In the testing environment testing the software and finding the bugs which present in developed software. This is access to testing team. In test lead and senior tester and testing engineer are there.

PRODUCTION ENVIRONMEN:

In the production environment which is access to the customer .when software is ready then install to the customer place in production server to use this run the business by customer.

WORK ALLOCATRION:

Work allocating to the developer or tester based on there work experience.

Example:

- critical work is given to 5-6 experience persons
- major work given to 3-2 experience persons.
- Miner work given to fresher 1 year experience.

BUILD PROCESS:

Business analyst will collect the requirement and given to project manager. He will give to development team & testing team.

→development team will develop the application and compile & compress the source code. White box testing is done. Then build is ready the development lead inform to test lead as

- Build is ready
- It is in server….(source code)
- Path: D drive/BO1
- Module is ready
- White box testing is ready.

→develop team develop the software that time testing will do understanding the requirement identify the scenarios and write test cases. Once development lead informs to test lead. Then test lead go to server which is given by development lead. copy the build (BO1) and install the software in test server and given URL to engineers the test the software.

In order to finding the bugs and report to development team as bug report tools like **zira , ALM**.

→testing team will testing the module in that time development team develop another module and bug fix which are reported by development team and again report to test lead.

→test lead again go to develop server copy the module of built install in test server. Before install the test server. We may delete old module so again retest first new module, bug fixes and last we will test old module. Finding bugs report to development team this process is called build process.

Why test engineer will find new bugs in the old modules ?

- Adding new module might introduce defect in old module.
- Fixing the defect might introduce mew defect in the old module
- Test engineer might have missed defects in the previous test cycle.

When do we do system testing?

- ⇨ when minimum bunch of modules are ready.
- ⇨ When the testing environment is like production environment is available.
- ⇨ When the basic functionalities are working fine.

What is release or first release or 1 release?

Starting from gathering the requirements followed by developing the software and testing the software. for many cycles and deploying the software into the production server is called release.

When to release the software to the customer?

- ❖ When all the features requested by the customer are ready
- ❖ When there are no blocker and critical bugs
- ❖ When the product is functionally stable
- ❖ Once after the software is tested in testing server like production server
- ❖ When all the end-to-end scenarios are working fine
- ❖ When we are about to meet deadline or release date given by customer

TEST CYCLE:

It is an effort or duration to start and complete the testing.

The test cycle duration can be 1 day or 2 days or 3 days its dependent on size of the application, complexity of the application and size of the testing team.

EX:

Project	1month=4units
Duration :1year	10months=40units
2months→write test cases	test cycle=5days for 1 unit
10months→test execution	

Re-spin:

Getting more than one built with in a test cycle is called re-spin.

Whenever tester find blocker bug developer will give re-spin.

Patch:

Set of modifying programs is called patch.

When there are blocker bugs developer will give patch.

Re-spin is referred as patch.

Build:

The process of source code file is convert into executed file knowns as build.

CONTINUOUS INTEGRATION:

Continuous integration (CI) is the practice of automating the integration of code changes from multiple contributor into a single software project . it is a primary develop best practice, allowing developer to frequently merge code change into a central repository where builds and tests then run.

CONTINUOUS INTEGRATION TOOLS:

- Jenkins
- Circleel
- Team city
- Bamboo
- Gitlab
- Travis CI
- Buddy
- Codeshoe

Who is involved installation of build?

⇨ Any body from testing team.
⇨ Any body from development team.
⇨ Build engineer or release engineer.

Version control tool (source code management tools):

Version control is a way to keep a track of the change in the code so that if something goes wrong. We can comparisons in different code version are revert to any previous version that we want. it is very much required where multiple developers are continuously working on/change the source code.

Best version control tool:

Git

CVS (concurrent version system)

SVN (apache sub version)

Mercuria

Monotone

Bazaar

TFS(team foundation server)

ACCEPTANCE TESTING:

Approach:1

It is an end-to-end testing done by it engineer sitting in customer place where in they take real time business scenarios and check whether software is capable of handling it.

Approach:2 UAT(user acceptance testing)

It is an end-to-end testing done by the end users where in they use the software for the business for a particular period of time and check whether software is capable of handling real-time business scenarios.

Approach no.3:

It is an end-to-end testing done by test engineer sitting in the customers place where in they check whether the software is capable of handling real time business scenarios.

Approach no.4:

It is an end-to-end testing done by test engineer in their own place where in they check whether the software is capable of handling real time business scenarios.

Approach no.5:

It is an end to end testing done by customers customers . where in they check whether the software is capable of handling real time business scenarios.

The no. of acceptance test cycle might increase because of the following reason: -

1. Test team is not proper testing.
2. Customer might ask more changes once after the software is delivered to the customer.
3. The requirement which was given in the beginning is not clear it is lead lot of reworks.

Alpha Testing	Beta Testing
Alpha testing performed by Testers who are usually internal employees of the organization	Beta testing is performed by Clients or End Users who are not employees of the organization
Alpha Testing performed at developer's site	Beta testing is performed at a client location
Alpha testing involves both the white box and black box testing	Beta Testing typically uses Black Box Testing
Alpha testing requires a lab environment or testing environment	Beta testing doesn't require any lab environment or testing environment. The software is made available to the public and is said to be real time environment
Long execution cycle may be required for Alpha testing	Only a few weeks of execution are required for Beta testing
Critical issues or fixes can be addressed by developers immediately in Alpha testing	Most of the issues or feedback is collected from Beta testing will be implemented in future versions of the product
Alpha testing is to ensure the quality of the product before moving to Beta testing	Beta testing also concentrates on the quality of the product, but gathers users input on the product and ensures that the product is ready for real time users

Smoke Testing:

Smoke Testing/Sanity Testing

/Build Verification Testing/

Dry run testing/

Confidence Testing

- Testing the basic and critical features of an application before going thorough testing is called as smoke testing.
- It is also called Build Verification Testing – because we check whether the build is broken or not.
- **How to do Smoke Testing?**
- Here list the features to be tested as part of smoke testing and list the features are not to be tested as part of smoke testing and test. In smoke testing, we do only positive testing – i.e, we enter only valid data and not invalid data.

Project: Gmail.

Features to tested as part of ST	Features not to be tested as part of ST
Signup, Login, Compose, Inbox, Sent Item, Draft, Search Mail, Reply mail, All Mails	Forgot pwd, Remember pwd, Starred, Important, label, spam, trash, calendar, help Contacts, chat, Archive, labels etc,...

In smoke testing, we do only positive testing – i.e, we enter only valid data and not invalid data.

How to write smoke testing scenarios:

1. To check that when user enter url, welcome page should be displayed.

2. To check that when user click on signup, signup page should be displayed

3. To check that when user enters valid data for all the fields in

the signup page and click on submit button account should be created

4.To check that user can login to application with valid login credentials.

When we do smoke testing:

1.As soon as we get new build from development team we should do smoke testing

2.Whenever the build comes to the customer, before the customer / client does Acceptance Testing, they also does Smoke Testing.

3. Release engg/ build engg will do smoke testing to check whether build is installed properly or not in testing server or production server

4. Developers will do smoke testing after doing WBT and before giving

build to testing team

Why we do smoke testing?

1. To check whether software is testable or not.

2. First day itself while doing smoke testing if you find defect

communicate to developer so that developer will get sufficient time

to fix the defect.

3. We do smoke testing to ensure that product is installed properly.

4. Developer is giving new build means he will be doing code changes,

chances are there this might affect old basic and critical features, so

in order to find that we do smoke testing.

5. It is like a health check of the product so smoke testing should be done

Types of Smoke Testing:

Formal Smoke Testing – the development team sends the s/w to the test lead. The test lead then instructs the testing team to do smoke testing and send report after smoke testing. Once, the testing team is done with smoke testing, they send the smoke testing report to the Test lead.

Informal Smoke Testing – here, the test lead says the product is ready and to start testing. He does not specify to do smoke testing. But, still the testing team start testing the product by doing smoke testing.

Smoke Testing	Sanity Testing
Smoke Testing is performed to check that the critical functionalities of the application is working fine	Sanity Testing is done to check the new functionality/bugs have been fixed
This testing is performed by the developers or testers	Sanity testing is usually performed by testers
Smoke testing is usually documented or scripted	Sanity testing is usually not documented and is unscripted
Smoke testing is a subset of Acceptance testing	Sanity testing is a subset of Regression Testing
Smoke testing exercises the entire system from end to end	Sanity testing exercises only the particular component of the entire system
Smoke testing is like General Health Check Up	Sanity Testing is like specialized health check up

Smoke testing can be done both manually and also with the help of automation tools. But the best and preferred way is to use automation tools to save time.

Advantages:

Easy to perform .

Reduces the risk.

Defects are identified at a very early stage.

Saves efforts, time and money.

Runs quickly if automated.

ADHOC TESTING:

AD-HOC testing also called monkey testing /gorilla testing.

Testing the application randomly is called ad-hoc testing.

Why we do ad hoc testing?

1) End-users use the application randomly and they may see a defect, but professional TE uses the application systematically so they may not find the same defect. In order to avoid this scenario, TE should go and then test the application randomly (i.e, behave like and end-user and test).

2) Development team looks at the requirements and build the product. Testing Team also look at the requirements and do the testing. By this method, Testing Team may not catch many bugs. In order to avoid this, we do random testing behaving like end-users.

3. In order to increase the bug count we should do adhoc testing

4. In Order to improve the test coverage we do adhoc testing.

5. In order to ensure the application is working according to implicit

requirement we do adhoc testing.

6.the intention of adhoc testing is to some ow break the product.

How to do adhoc testing.

1.Login to Gmail, click on compose, enter values for all the fields, click on send button, click on logout, click on browser back button and check whether login page is displayed.

2.Login to Gmail with valid login credentials copy the home page URL, click on logout, Open the browser and past the URL and check whether login page is displayed.

3. login to Gmail, click on compose and send mail, click on inbox, click on logout go to browser history click an inbox page related link and check whether login page is displayed.

4. Login to Gmail application using Mozilla and chrome browser, change the password in any one browser and check whether the application is logout from other browser.

When to do ad-hoc testing:

1. After testing as per requirement , the we start with ad-hoc testing.
2. When a good scenario comes ,we can stop function testing, integration testing, system testing and try that scenario for ad-hoc testing but we should not spend more time doing as-hoc testing and immediately resume with formal testing.
3. If there are more such scenarios , then we record it and do it at the last when we have time.
4. Whenever we are free, we do ad-hoc testing.

What are the types of ad-hoc testing:

1. BUDDY TESTING:

Here test engineer will sit with developer, discuss and come up with creative scenarios and test the software.

2. PAIR TESTING:

Here the test engineer will sit with other test engineer and discuss and come up with creative scenarios and test the application.

3. MONKEY TESTING:

Here the test engineer will test the application like a money without applying any logic. this is called money testing.

EXPLORATORY TESTING

Explore the application understand each and every feature, based on understanding identify all possible scenarios and document it, refer the documented scenarios and test the application is called Exploratory testing.

Drawback:

- Test engineer might misunderstand feature as defect and defect as feature.
- If any feature are not implemented test engineer may not be able to find it

Below are some tips/tricks that you can use in ET:

- Divide the application into modules and bifurcate modules into different pages.
- Start your ET from the pages. This will give the right coverage. Make a checklist of all the features and put a check mark when that is covered.
- Start with a basic scenario and then gradually enhance it to add more features to test it.
- Test all the input fields.
- Test for the error message
- Test all the negative scenarios.

REGRESSION TESTING:

Regression testing is a type of testing that is done verify that a code change in the software does not impact the existing functionality of the application.

Or

Testing the old features to make sure that change like adding new feature, deleting feature, modifying feature and fixing defects, are not introduced defects in the old feature is called regression testing.

Type of regression testing:

1. Unit regression testing.
2. Regional / partial regression testing.
3. Full/complete regression testing.

Unit regression testing:

Testing only the modified functionalities of the application is called unit regression testing.

Regional/partial regression testing:

Testing the changes along with the impacted modules of the application is called regional (or) partial regression testing.

How do you identify impacted or how do you do impact analysis or how do you pick regression test cases?

- Based on domain knowledge.
- By preparing impact analysis matrix.
- By conducting impact analysis meeting.

Testing team will do impact analysis meeting.

Test lead should communicate with development, team, testing, team, B.A, customer and prepare consolidated impact list.

Full regression testing:

Testing the changes along with the remaining modules of the application is called full regression testing.

when do we go for full regression testing:

- Whenever more changes made in the software
- When the changes made in the core features of an application.
- Before launching software to the customer the last few cycle we do full regression testing.

Note: doing functional ,integration and system testing on old feature is called regression.

Progression testing:

Testing new modules of the application is called progression testing.

Regression testing	**Re testing**
1)Testing the unchanged functionality of application to make sure that it is not broken because of changes.	1)Verifying whether the bug is fixed or not.
2)Defect verification is not part of regression testing.	2)defect verification is the part of retesting.
3)based on the project and availability of resources , regression testing can be carried out parallel with re-testing.	3)priority of re-testing is higher than regression testing, so it is carried out before regression testing.
4)you can do automation for regression testing.	4)you can not automated the testing cases for re testing.
5)regression testing is done for passed test cases.	5)retesting is done only for failed test cases.
6)regression testing check for unexpected side-effects.	6)re-testing makes sure that the original fault has corrected.

Software testing can be done in two ways.

1) Manual testing.
2) Automation (or) test automation

Manual testing:

Testing the functionality of the application manually using tools is called manual testing.

Automation testing (or) test automation:

Testing the functionality of application using tools is called automation testing.

Drawback of manual testing:

- Manual testing requires more time or more resources, some times both time and resources.
- Less accuracy.
- Performance testing is impractical in manual testing.
- Comparing large amount of data is impractical.
- Executing same tests again and again is time taking process as well as tedious.
- For every release you most rerun the same set of tests which on be tiresome.

Advantages of manual testing:

1) No environment limitation.
2) Programming knowledge is not required.
3) Recommendable for dynamically changing GUI design.
4) Manual testing allows for human observations which may be useful to find potential defects.

Regression testing tools:

- Selenium ,UFT
- Test compete
- Ranorex studio
- Katalon studio

- V test
- Watir
- Actiwate
- Rational functional tester.

Advantages of test automation:

1. Fast
2. Reliable
3. Reusable
4. Repeatable
5. Programmable

Drawback of automation:

- It is expensive.
- Lack of expertization
- Requires constant maintenance.
- We cannot automate all the areas (Either technology may not be supported or it will be very costly).

When to go for automation:

1. When there are more numbers of test cases.
2. When the product is functionally stable.
3. When there are no blockers and critical defects.
4. Once after the software is manually tested for 1 or 2 releases , then we go for automation.

Test scenarios:

The scenarios are the high-level documentation for all the customer business workflow according to the requirements is called test scenarios.

Test case:

It is a detailed document which cover all possible scenarios for a specific requirement.

What are the drawbacks of not writing test cases?

or

What will happen if you look at the requirement and test the application?

or

What are the drawback if you test the application by directly referring the requirement?

Solution:

- There is no consistency in the text execution.
- Test engineer might miss lot of defects.
- Quality of testing varies from person to person
- Testing depends on memory power of the test engineer.
- Chances are there we might test same scenarios again and again.
- Test coverage will not be good.

When do we write test case?

- When customer gives new requirement we should write test case.
- When customer wants to add new features or extra features we should write test case.
- When customer wants to do modification on the existing feature we should write test case.
- While testing the software if test engineer come up with creative scenarios we should write test case.
- While testing the software if test engineer finds any defect and if test case is not present for the defect then we should update test case for the defect.

Test case design technique / testing technique /design technique / black box technique / software design technique.

Test design techniques are the techniques which is applied while writing test cases in order to improve the test case coverage.

Error guessing:

Here we guess all the passible errors for the given application based on experience and intuition.

ECP (equivalent class partitioning)

1)pressman method:

Rule1:

If the given input if it is a range of value then derive the scenarios for 1 valid & 2 invalid values.

Rule2:

if the given input if it is a set of values then derive the scenarios for 1 valid & 2 invalid values.

Rule3:

If the given input if it is a Boolean then derive the scenarios for both true & false value.

2)practice method:

If the given input if it is a range of values divide the range into equivalent parts and test it for all the values but we should make sure that we are testing for at least two invalid values.

Note:

When there is a deviation in the given range we should prefer partice method.

BVA(boundary value analysis):

If the given input if it's a range of values between A and B the derived the scenarios for A,A+1,A-1 & B,B+1,B-1.

Test case template:

- In every company test engineer will write test cases in the test case template only.
- Test case template is not standard, it can varies from project to project and company to company.
- Test case template will be prepared in the test case management tool or MS-excel.

Test case review process:

Test case review process is an important process to follow in software testing. Test case ensures that each and every functionality mentioned in Software Requirement Specification is covered.

Test case should be effective and also follow the standards to write test case. To success and completeness of any test cases every test case should be reviewed. There are different types of test case review process.

Test Case Reviews can be done in three ways:

1. Self-review: It is done by the tester himself who has written the test cases. He can verify whether all the requirements are covered or not by looking into SRS/FRD.

2. Peer review: It is done by another tester who hasn't written those test cases but is familiar with the system under test. Also known as Maker and Checker review.

3. Review by a supervisor: It is done by a team lead or manager who is superior to the tester who has written the test cases and has great knowledge about the requirements and system under test.

While reviewing, the reviewer checks the following,

1) Template - he checks whether the template is as per decided for the project

2)Header:

a) Checks whether all the attributes are captured or not

b) Checks whether all the attributes in the header are filled or not

c)checks whether all the attributes in the header are relevant or not

3) Body:

a) Check whether all possible scenarios are covered or not

b) Check whether the flow of test case is good or not

c) Check whether the test case design techniques are applied or not

d) The test cases should be organized in such a way that it should less time to execute

e) Check whether the test case is simple to understand and execute

f) Check whether proper navigation steps is written or not

points to remember / tips while reviewing test cases:

1. While reviewing test case ,it is better to have a copy of SRS/FRD with you for the reference.
2. If you are not sure about any test case or expected results ,it is better to discuss with the client or your supervisor before making any decision.
3. If possible then try to execute test cases on the SUT(system under test) to have a better understanding of the results and execution steps.
4. It is always better to have face to face meeting with the tester to make him understand all the review feedback properly.
5. It is recommended to follow version numbers in review process.

procedure to write the test case:

SYSTEM STUDY

IDENTIFY SCENARIOUS

PRIORITIZE SCENARIOUS

WRITE TEST CASES

REVIEW TEST CASES

FIX THE REVIEW COMMENTS

VERIFY THE FIX

APPROVE TEXT LEAD

STORE THE TEST CASES IN THE TEST CASE REGORSITY

Brainstorming meeting

It is a meeting conducted by testing team, in this meeting they will discuss about scenarios what they have identified, here each and every test engg will explain the scenarios what they have identified, where in other TE will try to understand the scenarios what he is explaining.

If the other test engg finds any defects in the scenarios like wrong, missing, duplicate scenarios that will be given as a feedback to the test engg.

The test engg will update the scenarios, This cycle will repeats on every test engg.

This meeting will be measured by TL once after the meeting

With the help of this meeting we will have a good coverage in the scenarios.

How do you ensure that your test coverage is good? Or how do you convince your test manager or customer standing your test coverage is good?

1. have written test case by applying test case design technique, so my test coverage is good.

2. I have covered all positive and negative scenarios

3. I have conducted brainstorming meeting, so my lest coverage is good

4. I have got my test cases reviewed by other test engg.

5. I have written test cases by following procedure to write test case method 6. While test execution found creative scenarios and added into test case

7. I will do adhoc testing and this will help me to improve test coverage

8. I have spent more time in doing system study, because of that i got very good project knowledge which helped me to write more no. of scenarios

9. I have prepared traceability matrix and ensured each and every requirement has at least one test case, so my test coverage is good

STLC

(software testing life cycle)

- STLC stands for software testing life cycle.
- It is standard procedure to test the software.
- It is a part of software development life cycle (SDLC).
- It has different stage.

SYSTEM STUDY

WRITE TEST PLAN

WRITE TEST CASE

TRACEBILITY MATRIX

TEST EXECUTION

DEFECT TRACKING

TEST EXECUTION REPORT

RETROSPECTIVE MEETING

System study:-

Here tester should try to understand requirement if they have any queries they should talk to the one who known the requirement very well and clarify the queries.

Test Plan:

Test plan is a document which derives all future activities of the project.

All future testing activities is planned and put into a document and this document is known as Test Plan. It contains -number of engineers needed for the project, who should test which feature, how the defects must be communicated to the development team, when we should start and finish writing test cases, executing test cases, what are the types of testing we use to test for the application etc.

Write test case:

We write test case for every feature, these test cases are reviewed, and after all mistakes are corrected and once the test cases are approved. Then they stored in the test case repository.

Traceability Matrix

Traceability Matrix - it is a document which ensures that every requirement has a test case.

Test cases are written by looking at the requirements and application is tested by executing test cases. If any requirement is missed i.e, test cases are not written for a particular requirement, then that particular feature is not tested which may have some bugs. Just to ensure that all the requirements are converted. traceability matrix is written.

The Traceability Matrix is also known as **RTM** (Requirement Traceability Matrix) or **CRM**(Cross Reference Matrix).

Advantages of Traceability Matrix

1. Test Coverage will be good

2. We will get to know which feature should be tested manually and Automatically

3. Whenever the requirement changes we will get to know which are the test cases and automation scripts needs to be change.

Types of Traceability Matrix

1. **Forward traceability matrix:** It maps requirements to test cases.

2. **Backward traceability matrix**: It Maps the test case to the requirement

3. **Bidirectional traceability matrix:** This traceability matrix ensures that all requirements are covered by test cases.

Traceability Matrix Template:

It has got following fields.

SL No.

Module Name

High Level Requirement

Detailed Requirement

Test case name

Test Script name

Execution type

Achievements:

- Test case were reviewed by the peers and got many inputs because of this my test coverage is good.
- We did brainstorming meeting which helped us to improve scenarios Coverage
- We did impact analysis meeting which helped to identify impacted areas.
- we swapped modules between the test engineer and did testing which helped us to find more no. of defects.

Retrospect meeting(also called Post Mortem Meeting/Project Closure Meeting)

The Test Manager calls everyone in the testing team for a meeting and asks them for a list of mistakes and achievements in the project

This is done by test lead or test manager. Here, the manager documents this retrospect meeting and stores it in QMS (Quality Management System). It is a folder, where inside this folder, there is another folder called Retrospect folder and here this excel sheet document is stored. When we get new project, while we write the test plan- we will open this retrospect file and will try and implement the good and correct the mistakes

Mistakes

- Dev team didn't fix the blocker defects on time which delayed our testing
- Requirements were not clear
- TC Review did not happen properly.

Test execution:

Test execution is the process of execution the test case and comparing the expected and actual results.

Defect tracking:

In this section we mention how to communicate. The defects found during testing to the development team should respond to it.

Test execution report:

Test execution report is prepared after every test cycle and sent to development team, testing team, management and customer (depends if it is a fixed big project is over – according to the customer.

Defect life cycle

What is a defect?

It is a mismatch in the expected and actual behavior of an application.

Or

It is a deviation from the customers requirement specification.

Example:

1. Unable to create account even after entering valid data to all the required fields.

2. Added products are not displayed in the cart.

Defect occurs in the application because of the following reasons.

Wrong implementation: On click compose link, inbox page is displayed

Missing implementation: Sent item module is not exists in the application

Extra implementation: Links provided to another applications.

What is the major difference between Error, Defect/Bug, and Failure?

Error: If the developers find that there is a mismatch in the actual and expected behaviour of an application in the development phase then they call it an Error.

Defect: If testers find a mismatch in the actual and expected behaviour of an application in the testing phase then they call it as a Defect.

Failure: If customers or end-users find a mismatch in the actual and expected behaviour of an application in the production phase then they call it a Failure.

Defect Report

While reporting the bug to developer, your Bug Report should contain the following information

Defect_ID - Unique identification number for the defect.

(It will be automatically created by the BUG Tracking tool once you save this bug)

Release Name: Here we should specify the name of the release

Build Id: Version of the application in which defect was found.

Module Name: Specific module of the product where the defect was detected.

Status: New/Assigned/open/fixed/reopen/closed (Depends on the Tool you are using)

Severity: Blocker/Showstopper

- Critical
- Major
- Minor

Priority: Urgent

- High
- Medium
- low

Test Environment: Here we should specify test environment details like the operating system, Browser used etc...

Test data: The data used to find the bug.

Brief Description: Here we should specify the summary of the defect

Detailed Description: Here we should specify the steps to reproduce the defect

Expected Result:

Actual Result:

Attachment:

Author Name:

Created date:

Defect Life Cycle or Bug life Cycle

Defect life cycle or Bug Life Cycle is the specific set of states that a bug goes through in its entire life. This starts as soon as any new defect is found by a tester and comes to an end when a tester closes that defect assuring that it won't get reproduced again.

Bug Life Cycle Status

The number of states that a defect goes through varies from project to project. Below lifecycle diagram, covers all possible states.

New: This is the first state of a defect in the Defect Life Cycle. When any new defect is found, it falls in a 'New' status,

Assigned: In this stage, a newly created defect is assigned to the development team for working on the defect. This is assigned by the test lead or the manager of the testing team to a developer.

Open: Here, the developer starts the process of analysing the defect and works on fixing it, if required. If the developer feels that the defect is not appropriate then it may get transferred to any of the below four states namely Duplicate, Deferred, Rejected, or Not a Bug-based upon specific reason.

Fixed: When the developer finishes the task of fixing a defect by making the required changes then he can mark the status of the defect as 'Fixed'.

Pending Retest: After fixing the defect, the developer assigns the defect to the tester for retesting the defect at their end, and till the

tester works on retesting the defect, the state of the defect remains in 'Pending Retest'.

Retest: At this point, the tester starts the task of working on the retesting of the defect to verify if the defect is fixed accurately by the developer as per the requirements or not.

Reopen: If the bug persists even after the developer has fixed the bug, the tester changes the status to "reopened". Once again the bug goes through the life cycle.

Verified: The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is "verified."

Closed: If the bug is no longer exists then tester assigns the status "Closed."

Duplicate:

If the developer finds the defect as same as any other defect or if the concept of the defect matches any other defect then the status of the defect is changed to 'Duplicate' by the developer.

Reason for duplicate:

- Common features
- Dependent features

How to avoid duplicate bugs?

As soon as bug found search in bug repository, if the bug is exist don't report the bug.

Rejected:

Now, when the TE sends a defect report - the Development Lead will look at it and reject the bug.

Bug is rejected because,

1. Misunderstanding of requirements

2. Referring to old requirements

3. Because of extra features

4. Improper installation of the Product

Test engineer Approach:

Whenever the developer says invalid, first we reconfirm by going through the requirement once again and update the status accordingly. i.e.

- **Close:** If really the bug is invalid
- **Reopen:** If the bug is valid

Deferred or Postponed

If the developer feels that the defect is not of very important priority and it can get fixed in the next releases he can change the status of the defect as 'Deferred'.

Why do you get postponed status?

- If We find a bug during the end of the release (could be major or minor but cannot be critical) - developers won't have time to fix the bug - such a bug will be postponed and fixed in the next release and the bug status will be "postponed"
- If the tester finds a bug in a feature where in developers are expecting changes from customers side those defects will get the status postponed.
- If the bug is minor and it is in feature exposed to internal users.

Cannot be fixed:

Chances are there -test engineer finds a bug and sends it ti development lead -development lead looks at the bug and sends it back saying “cannot be fixed”.

⇨ Technology itself is not supporting i.e, programming language . we are using it self is not having capability to solve the problem.

- ⇨ Whenever there is a bug in the root of the product if it's a minor bug, then development lead says "cannot be fixed ". But ,if it's a critical bug , then development lead cannot reject the bug.
- ⇨ If the cost of fixing the bug is more than the cost of the bug itself – cost of the bug mean loss incurred because of the bug.

Test engineer approach:

If the status is given as cannot be fixed then test engineer checks whether if it is blocker and critical bug, if it's a blocker or critical bug then the test engineer will change the status of the bug to reopen if it is a minor bug test engineer will given the status as closed.

How to avoid cannot be fixed status as a test engineer.

As a test engineer we cant avoid this status of the bug as we nothing to do with the technology or if any problem in the core of the code.

Not reproducible:

Test engineer are able to see the defect bot the same defect is not able to see by the dev team. Then development lead will change the staud of the bug as not reproducible.

Why we get "not reproducible " defects?

- Because of platform mismatch.
- Because of improper defect report.
- Because of data mismatch.
- Because of build mismatch.
- Because of inconsistent defects.

A defect that is not occurring repeatedly in every execution and is producing only at same instances and whose steps as proof have to be captured with the help of screenshot, then such a defect is called as a "inconsistent" defect.

Request for enhancement(RFE):

Test engineer finds a bug and sends it to development team – when development team look at report sent by the engineer – they known

it’s a bug, but they say its not a bug because its not part of the requirement.

Example for RFE

- ⇨ To add clear button in create customer page.
- ⇨ To add select all checkbox in approve loans page of banking app.
- ⇨ To provide future scheduling option in appointment module of health care application.
- ⇨ To provide scheduling message in whatsapp.
- ⇨ To provide auto back up functionality in whatsapp.
- ⇨ To provide option to sync whatsapp contacts with google contacts.

compatibility testing:

testing the functionality of an application in different software and hardware environment is called compatibility testing.

Why we do compatibility testing:

1. To check whether software work consistently in all the platforms.
2. Usually developing team and testing team use the same platfrom while developing the software , but once after the applicaton is released in the production server the customer may find bugs in the application and hence we do compatibility testing.

Type of software compatibility testing:

- Browser compatibility test.
- Hardware compatibility testing.
- Network compatibility testing.
- Mobile devices compatibility testing.
- Operating system compatibility testing.

Hardware:-

It is to test the application /software compatibility with the different hardware configuration.

- ⇨ Test on different processers.
- ⇨ Test different ram.
- ⇨ Test on different motherboard.
- ⇨ Test on different VGA card.

➔version history of android **OS.**

➔version history of **IOS.**

➔version history of **windows OS.**

Network:

it is to check the application in a different network like 3G, 4G, 5G wifi, etc.

Mobile devices

It is to check if the application is compatible with mobile devices and their platform like android, ios, windows, etc

Software (operating system):

It is to check if the application is compatible with different operating system like window, linux, MAC, etc

when do we perform compatibility testing?

When the application is relatively stable on the base platform me perform compatibility testing

How to do compatibility testing?

Will execute manual test cases as automation scripts fer platforms and do compatibility testing.

Compatibility testing:

- Scattered content
- Alignment issues
- Broken frames.

- Change in look and feel of the application
- Object overlapping.
- Change in font size, style, colour.

PERFORMANCE TESTING:

Verifying the stability & response time of an application by applying load is called as performance testing.

1. Stability: ability of an application to respond to the users (efficiency)

2. Response time: over all time taken by the application to respond to the user.

3. Load: number of users using the application at a time.

Tools used in performance testing

1.J meter (java heter)

2. Load runner

3.Neo load

4. Silk perfower

All the above tools are licensed

1. Load testing: verifying the stability and response time of an application by applying less number of load and equal number of loads

2. Stress testing: verifying the stability and response time of an application by applying more number of load than the designed user. In stress testing testers verify three conditions

1. If the application is still working

2. If the application has crashed

3. Is it displaying error message accordingly, if any of these conditions are missing then raise it as a defect.

3.volume testing:

Verifying the stability & response time of an application by transferring huge volume of data from once place to another place is called as volume testing.

Soak testing:

Testing the stability and response time of an application by applying lead for a continuous period of time is known as soak testing.

Recovery testing:

Verifying how fast the application recovers back from a crash, disaster, failure without any data loss to the original position is called as recovery testing.

Ex: application→ chrome

10mb download

5mb

Internet connection loss(disaster/crash)

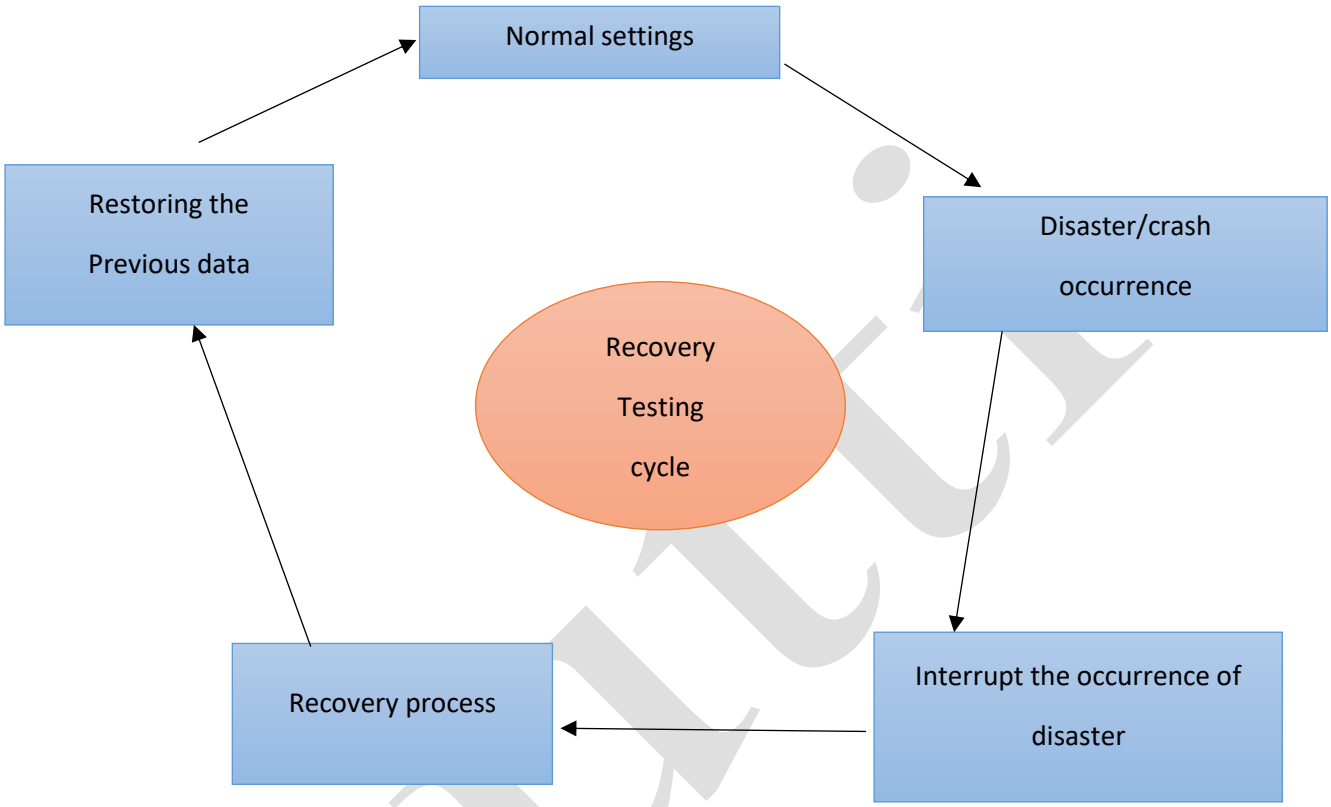
Interact connection resolved continue.

⇨ Only trained test engineers will perform recovery testing because they know how to crash the application and also how to recover back to the normal stage.

How to do recovery testing:

- The trained test engineer will create a crash or disaster by applying forced defects and check if the application is failing at any particular point.
- Here they verify error log message with instruction (if it is not displayed raise it as defect)

- The test engineer will end the process which has caused the defect and verify if the application is recovering back to the pervious setting without ant data loss.



Reliability testing:

Testing the functionality of an application for a continuous period of time is known as reliability testing.

Reliable:-

Why we do reliability testing:

- To check whether the application is failing at any point
- To identify why is it failing.
- How many times is it failing.

How to do reliability testing:

We use automation tools where the testers write the specific scripts for the tool to check the application if it is failing at any point of time.

Same of the tools are:

1. Weibull++ - reliability data analysis
2. RGA - reliability growth analysis
3. RCM – reliability cantered maintenance.

When to do reliability testing:

When the application is stable we perform reliability testing.

Some of the real time examples are:

Social media application : facebook ,Instagram ,whatsapp etc.

Gaming application : PUBG,temple run etc:

Media entertainment : hotstar ,reflex etc.

Education domain : BYJU's

Note: it is not a mandatory testing and is done based on the customer domain and the type of application.

Priority and severity:

severity is a impact of a defect on the customer business the different severity levels are.

Blocker:-

This defect indicates complete shot down of the process , nothing can proceed further.

Critical:

It is a highly severe defect and collapse the system. However ,certain parts of the system.

Major:

It causes some considerable behaviour, but the system is still functional .

Minor:

it wont cause any major break down of the system.

Priority:

It is defined as the order in which a defect should be fixed higher the priority the sooner the defect should be resolved.

The different priority levels are.

High: the defect must be resolved as soon possible as it affects system severely and cannot be used until it is fixed.

Medium :

during the normal course of the development activities defect should be resolved it can wait until new version is created.

Low:

The defect is an irritant bot repair can be done once .the serious defect has fixed.

High severity and high priority

1.place order functionality is not working in online shopping application

2. Unable to select current location and destination location

is ola application

3.join meeting option is not working in skype application

4.send request option is disable in facebook application

5.send message feature is not working in whatsapp

6. The user performs adding an item to the cart, the number of quantities added is incorrect / wrong product gets added

Low severity & low priority

1.spelling mistake in the confirmation message

2.alignment issues in the achieved project page

3.confirmation message is not displayed after logout from gmail application

4.relatively more time is taken to terminate session in the go to meeting application

5.color of the text or tab is too dark

6.font size is small

High severity and low priority

1.blank page is displayed on click help link

2.external link provided in the application is not working

3.unable to install whatsapp application for 30th time

4.application crash on multiple click on delete mail button in trash module of the gmail.

Low severity & high priority

1.spelling mistakes in the name or logo

Test plan:

Test Plan is a document which derives all future testing activities of the project.

It has following sections:

1. Objective
2. Scope
3. Approach
4. Testing methodologies

5. Assumption
6. Risks
7. Backup plan
8. Schedule
9. Roles and responsibilities
10. Defect tracking
11. Test environmental
12. Entry and exit criteria
13. Test automation
14. Deliverables or test artifacts
15. Templates
16. Effort estimation

Objective:

It gives the aim of preparing test plan i.e why are we preparing this test plan.

Scope:

In the planning stage, we decide which feature to test and which not to test due to the limited time available for the project.

2.1 features to be tested
2.2 features not to be tested

Features not to be tested

a) "HELP" is a feature developed and written by a technical writer and reviewed by another technical writer. So, we'll not test this feature.

b) Third party modules of the application.

c)The application might be having link to some other application. Here, our scope of testing is limited to,

Whether link exists

If it goes to homepage of the corresponding application when we click on the link.

TESTING METHODOLOGIES

Depending upon the application, we decide what type of testing we do for the various features of the application.

4)APPROACH

The way we go about testing the product in future,

a) By writing high level scenarios

b) By writing flow graphs

5)Assumptions:

When writing test plans, certain assumptions would be made like technology, resources etc.

6) RISKS

If the assumptions fail, risks are involved

7) CONTINGENCY PLAN OR MITIGATION PLAN OR BACK-UP PLAN

To overcome the risks, a contingency plan has to be made. At least to reduce the percentage from 100% to 20%

Always assumptions, risks, mitigation plan are specific to the project.

The different types of risks involved are,

- Resource point of view
- Technical point of view
- Customer point of view

8) SCHEDULES:-

This section contains - when exactly each activity should start and end?

9) ROLES AND RESPONSIBILITIES

9.1 Test Manager

Ø.Writes or reviews test plan

Ø.Test manager will interact with customer, management, development team and testing team

Ø.Sign off release note

Ø.Handle issues and escalations

Ø.Test Manager will be involved in the effort estimati

Ø.Approve test case (not always)

Test Engineer 1

Ø.Involved in system study.

Ø.Involved in identifying scenarios.

Ø.Conducting brainstorming meeting and updating scenarios

Ø.Converting test scenarios into test cases

Ø.Involved in reviewing test case

Ø.Giving review comments and fixing the review comments

Ø.Involved in execution the test case

Ø.Involved in identifying the defects and communicate defects to dev team using defect tracking tool.

Ø.Involved in tracking the defects

Ø.Involved in selecting test case for regression testing

Ø.Involved in updating test case, whenever req are getting changed

ØInvolved in conducting bug triage meeting

Ø.Involved in performing traceability matrix

Test Engineer: (Automation Engg.)

Ø.Set up and install the product

Ø.Identify test cases to be automated

Ø.Automate identified test cases using Automation tool like Selenium, UFT, etc...

Ø.Execute and maintain automation scripts

10) DEFECT TRACKING

In this section, we mention - how to communicate the defects found during testing to the development team and also how development team should respond to it.

This section contains

10.1 Procedure to track the defects

10.2 Severity Level

10.3 Priority Level

10.4 Defect tracking tool to be used

11. Test Environment

In this section we mention what are the hardware and software needs to be used on order to set up test environment

11.1 Hardware:

Server Side: Server:- Sun Starcat 1500

Client: Processor: Intel 2GHz

RAM: 8GB

Hard disk: ITB

11.2 Software:

11.2.1 Server

OS: Linux

Web Server: TomCat

Application Server: Websphere

Database Server : Oracle (or) MS-SQL Server

11.2.2 Client

OS: W7, W8, W10

Browser: Chrome, Firefox

11.3 Procedure to install the software

12) Entry and Exit Criteria

Entry Criteria for FT:

a) WBT should be over

b) Test cases should be ready

c)Product should be installed with proper test environment

d) Test data should be ready

e) Resources should be available

Exit Criteria for FT:

It is decided based on

1)Based on %age test execution

Entry criteria for IT:

- should have met exit criteria of FT

(remaining all are same as entry criteria of FT)

Exit criteria for IT:

All points are same as exit criteria for FT.

But if the age pass for FT is 85%, then the %age pass for IT should be 90% - because as we reach the later stages of testing, we expect the number of defects to be less.

Entry criteria for ST :

- exit criteria of IT should be met

- minimum set of features must be developed

- test environment should be similar to production environment

(remaining all are same as of IT)

Exit criteria for ST :

- everything remains same as of above, but the pass % is now 99% - there should be 0 critical bugs. There could be some 30major and 50minor bugs. If all this is met, then product can be released.

13) TEST AUTOMATION

13.1 Features to be automated

13.2 Features not to be automated

13.3 Which is the automation tool you are planning to use

13.4 What is the automation framework you are planning to use

14.DELIVERABLES

It is the output from the testing team. It contains what we will deliver to the customer at the end of the project. It has the following sections,

v 14.1 Test Plan

v 14.2 Test Cases

v 14.3 Test Scripts

v 14.4 Traceability Matrix

v 14.5 Defect Report

v 14.6 Test Execution Report

v 14.7 Graphs and Metrics

14.7 Release Note

Release Note is a document prepared during release of the project and signed by test manager

The release note contains,

1) List of pending/open bugs

2) List of features added, modified or deleted

3)Platforms(OS, Browsers, Hardware) in which the product is tested

4) Platforms in which the product is not tested

5)List of bugs fixed in current release which were found in previous release production

6) Procedure to install the software

7) Version of the software

15 TEMPLATES

This section contains all the templates for the documents which will be used in the project.

The various documents which will be covered in the Template section are,

Test Case

Traceability Matrix

Test Execution Report

Defect Report

Test Case Review Template

16.Effort Estimation

Effort estimation is the process of predicting the most realistic amount of effort (expressed in terms of person-hours or money) required to develop, testing and maintain software.

To find total engineers required for the project

Total Engg=Total working days/Estimated days

Effort Variance=(Estimated Effort - Actual Effort/
Estimated Effort)*100

Globalisation testing:

Developing the software to support multiple language is called globalization.

Testing the software which is developed to supports multiple languages is called globalisation testing. There are two types of testing.

1.internationalistion testing (I 18 N testing)

2.localistion testing (L 10 N testing)

Internationalization testing

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

Here we also check the content is displayed in right position or not.

We add prefix to the property file in order to check the content is displayed in right language.

He add suffix to the property file in order to check the content is displayed in right position or not.

Adding prefix &suffix to the property file is called pseudo code translation

Localization testing

Here we check whether the content is changing locally according to country standards or not

ex: currency format, date format, time format, phone number format, pin code format

White box testing

Testing each and every line of the program is called white box testing. It is done by developers before giving the software to the testing team. In order to reduce the bugs in the software.

Types of wbt

1. Path testing

2. Condition testing

3. Loop testing

CONDITION TESTING

Here developers will test all the condition given in the source code for both true and false conditions

If (condition)

{

True

}

Else

{

False

}

LOOP TESTING

Here developer should test the loops for all the iteration. It can be done manually or automatically (unit test scripts)

WHILE (I<-5000)	TEST SCRIPT
{	{
----------	----------
----------	----------
}	}

Unit test scripts are written for checking loops and other programs automatically.

unit test scripts can be written for doing path, conditional, loop testing

PATH TESTING

Here developers should test each and every independent path of the application, test all the independent paths whenever the changes made in the source code, we should check for all the dependent paths.

NOTE

- Wbt testing can be done automatically or manually with the hel test programs (unit test scripts)
- a set of unit test scripts are known as unit test suit/ unit test case/ unit test list.

Agile model

What is Agile?

It is a standard procedure or step by step procedure to develop the new software.

Its an Iterative and incremental Approach.

PRINCIPLE OF AGILE METHODOLOGY

- The main goal of agile is customer satisfaction by quick delivery of working software.
- Customer can change the requirement at any point of development stage
- Release should be short
- There will be good communication between customer, Product owner, developer and Test Engineer
- Developers and test engineers will be doing lot of meetings at regular intervals in order to improve the quality of process

What is agile testing?

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

Types of agile methodologies?

1. Scrum methodologies

2. XP (Extreme Programming)

3. FOD (Feature driven development).

4.Crystal Clear

5. Lean and Kanban

6.ASDM (Adoptive software development method)

7.DSDM (Dynamic Software Development method)

Terminologies:

- Release
- EPIC
- User stories/stories/story card
- story point
- sprint
- sprint planning
- scrum master
- scrum meeting
- sprint retrospective meeting?
- Release retrospective meeting?
- Bug triage meeting
- Product backlog meeting

Scrum methodologies/Scrum Process:

Release: Combination of sprints is called release

EPIC: It is a larger requirement. Epics need to be broken into smaller deliverables (stories). Or we can also defined as complete set of requirement is called EPIC

One EPIC consist of multiple stories

User Stories/Stories/Story Cards:

A requirement that the business wants. It is something that is deliverable within a single sprint(Smaller pieces of requirement). It can be a feature or modules or functionalities.

Story Point:

It is rough estimation given developer and test engineer to develop and test individual stories

Ex: 1 story point-9 hours

1 SP-6

1 SP-8

Story point should be in Fibonacci series

QA Story points estimated based on the following aspects:

- Time spent on understanding story
- Write test scenarios
- Write test cases
- Review test cases
- Test case execution
- Defect tracking

Developer Story points estimated based on the following aspects:

- Understanding requirements
- Design
- Coding
- Reviewing code
- WBT
- Time spent on fixing the defects

Sprint:

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

Sprint planning:

Sprint planning is an event in scrum that defines what can be delivered in the upcoming sprint and how that work will be achieved.

Sprint planning is a meeting conducted by scrum master on the first day every sprint

Explain the requirement

Identify list of task to be done

Assign task to engineer

Scrum Meeting:

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

1. It is also called as daily stand up meeting
2. This meeting is strictly bounded for 15 minutes
3. In this meeting we discuss below mentioned points
1. What you did yesterday
2. What have you planned today
3. Are there any obstacles or impediments

Note: During a sprint, the team checks in during the dally scrum, , about how the work is progressing. The goal of this meeting it to surface any blockers and challenges that would impact The teams ability to deliver the sprint goal.

sprint review meeting

After a sprint, the team demonstrates what they've completed during the sprint review. This is your team's opportunity to showcase their work to stakeholders and teammates before it hits production.

Sprint retrospective meeting:

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

In this meeting we discuss the following points

1. What went well
2. What didn’t go well
3. Are there any actions plan

Release retrospective meeting:

it is the meeting conducted by scrum master on the last day of every release.

Bug triage meeting:

- This is the meeting conducted by the test engineer or scrum master a week or 2 week before the release
- BA, TE, Developer and SM will be a part of meeting
- In this meeting test engineer list all the open and pending bugs which are not fixed by the developers in the current and previous release as a team we will re prioritize the defect from the customers' business point of view and decide how many bugs should be fixed as part of a current release and how many bugs can be move to upcoming release.

product backlog meeting/ backlog grooming /backlog refinement:

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

- In this meeting test engineer and developer come up with list all the pending stories which are not implemented as part of the current and previous release.
- As a team we will re prioritize all the stories from the business point of view and move the stories to the next upcoming sprint or releases.

Resume points:

- Excellent knowledge about on agile scrum development process
- Very good knowledge on sprint planning meeting
- Very good knowledge on scrum meeting
- Very good knowledge on sprint retrospective meeting

Explain agile process which you are following

- In our scrum team we have scrum master (SM), Product owner (PO), developers and testers. .
- PO will create the epics and stories and they will be kept in backlogs.
- Twice in a week we will have the grooming call and in the call, PO will explain the requirement and based on that we will estimate the story points.
- Before we start the sprint we will have the sprint planning meeting and during this meeting we will pick the stories which we are going to work in the upcoming sprint.
- From the starting of the sprint developers will start coding for the features and we (testers) will create the test cases and the test data for the feature.
- Once the development work is completed, developers will deploy the code to Test environment and then we will execute the test cases.
- If any test case is failed then we will raise bug and assign to developers.
- During the sprint, we will have daily stand up call / scrum call in which we will discuss,

1. What did I do yesterday? o What I'm going to do today?
2. Bugs/Blockers and Impediments

- We are following 2 weeks sprints in which 1st 7 days are for development and functional testing and day 8, 9 and 10 are for regression testing.
- Once after the sprint ends we will have sprint retrospective meeting in which we will discuss,

What went well?

What went wrong?

What could have done better?

Type here to search

Different between agile and waterfall model:

s.no	WaterFall	Agile
1	The waterfall methodology is sequential and linear.	Agile methodology is incremental and iterative.
2	Requirements have to be frozen at the beginning of SDLC.	Requirements are expected to change and changes are incorporated at any point.
3	The working model of software is delivered at the later phases of SDLC.	The working model is delivered during the initial phases and successive iteration of the model is delivered to the client for feedback.
4	It is difficult to scale-up projects based on waterfall methodology.	Scaling up of products is easy because of the iterative approach.
5	Customers or end-user doesn't have a say after the requirements are frozen during the initial phases. They only get to know the product once it is built completely.	Frequent customer interaction and feedbacks are involved in agile methodology.
6	Testing is performed once the software is built.	Continuous testing is performed during each iteration.