Software testing

Why software testing?

Every software is developed in order to support the business, if there is a bug in the software it affects the business, so before we launch the software to the business it must be tested and the problems must be recognized and solved.

We do this to check whether the product is working according to the CM requirement

We do this to improve the quality of the product or s/w

Manual testing

Automation testing

What is software?

Software is a set of instructions, data and program that tells a computer how to work.

What is error?

Error is the mistake done in a code because of that we are not able to compile or we are not able to run

There are two types of error

Compile time error : we get compile time error because of mistakes in the syntax

Runtime error : we get runtime error because of logical mistakes

What is defect?

Defect is derived from the required specification.

What is bug?

Bug is an informal name given to the defect.

What is failure?

Defect leads to failure or defect causes failure.

#### What is Service – based companies and Product – based companies

##### Service – based companies: -

They provide service and develop software for other companies

They provide software which is and specified as per the client company‟s requirement and never keep the code of the developed product and does not provide the software to any other company other than the client company.

Ex – Wipro, Infosys, TCS, Accenture

##### Product – based companies:-

The develop software products and sell it to many companies which may need the software and make profits for themselves

They are the sole owners of the product they develop and the code used and sell it to other companies which may need the software.

Ex – Oracle, Microsoft

Software Development Lifecycle (SDLC)

It is a procedure to develop the software.

It is a process of creating or altering systems and the models and methodologies that people use to develop these systems. Any SDLC should result in a high quality system that meets or exceeds customer expectations, reaches completion within time and cost estimates, works effectively and efficiently and is inexpensive to maintain and cost effective to enhance.

Different procedures / models are available to develop software namely,

Waterfall model

It is a traditional model

It is a sequential design process, often used in SDLC, in which the progress is seen as flowing steadily downwards (like a waterfall), through the different phases as shown in the figure,

Why dev should not test the s/w?

* He is always in over confidence that whatever they build will always works
* They will never see the product from –ve point of view
* Developer somehow they want to build the product they don’t want to see the product breaking
* Even though they know that bugs are there they will hide it
* Chances are there to use the time allocated to testing to build the product and they say time is not there

Why requirements keep changing in many of the projects?

* Every software is developed to support the business and the business keps changing so customer requirements keep changing
* Because of competition requirements keeps changing
* To adopt the new technology

Why we will not allow requirement changes in this model?

* As the requirement changes design changes and when this design changes lot of mistakes may happen in the design
* A lot code will changed when the code changes lot of mistakes will happen in the code, to avoid that we frees the requirement.

Advantages of the water fall model

* It is simple to adopt
* Since, requirements are freezed at the end we get the stable product
* Initial investment is less

Draw back

* It is not flexible model
* Testing is small phase which is done only after coding
* Design is not tested and the requirement is not tested if there is a defect either in the requirement or in the design it will flow till the end and it leads to lots of re-work
* Total investment will be more because of rework

Application

* Whenever developing the small s/w or application
* Whenever we go for short term projects
* Whenever we are sure that the requirements are not going to change

Prototype Model

Static -

Dynamic

Hybrid model

Hybrid of spiral and prototype models

Hybrid of V and prototype models

Interview questions

What is SDLC?

Tell what

Tell different stages and explain each stages

Tell about different types of model

Explain each model

What are the different models available

Water fall model

Spiral

V

Prototype

Hybrid

Azile

What are the advantages, disadvantages and applications of each model

Can you explain waterfall model in detail

Can you explain v model

Can you explain Azilemodel

Project specific

In the current project which model you are following, can you explain

Tell Azile model, and explain it

Tell me your roles and responsibilities

How do you rate yourself in Manual testing/java/SQL/Selenium?

out of 5

7.5 out of 10

HR question

Please tell me about yourself / introduce yourself/ brief me about yourself

Why should we hire you?

Why S/W testing?

Why you want to join our company?

Where you want to find yourself down the line 5 years

Strengths and weakness

How much salary

Ready to relocate

Are you ready to work in shift

Software Testing

Software testing

Procedure of finding or identifying defects in the s/w is called s/w testing

It is verifying the functionality of the application against requirement specification.

It is the execution of the s/w with the intention of finding defects.

It is checking whether the s/w works according the requirements.

Types of testing

White box testing

Black box testing

White box testing

Will be done by dev E

We have 3 types White box testing

Path testing

Condition testing

Loop testing

What is WBT

Can you tell what are the types of WBT

Who dose the WBT, can you explain how they have done WBT?

Our developer have write the code and they test each and every line of the code, before they give the build to the testing team they run test suit if the majority of the tests are passed then only they give build to the testing team

Difference between WBT and BBT

Black Box Testing (BBT)

It is verifying the functionality or the behavior of the s/w against the requirement specification. This will be done by test engineers.

Types of Black Box Testing

Functional testing

Integration testing

System testing

Acceptance testing

Smoke testing

Adhoc testing

Internationalization testing

Localization testing

Exploratory testing

Regration testing ……………..

Functional testing / component testing / field level testing:

Testing each and every component thoroughly (rigorously) against requirement specification is known as functional testing.

Login as A – apply for OD RS 20,000- click on apply – logout

Login as Manager – Approve the OD of A – logout

Login A – check for OD balance – Rs 20,000 should be deposited – logout

Change the server date and to next 30 days.

Login as A – Check fo OD balance – 20,000+400+250=20650 – logout

Login as A –and click on repay – pay 20650- and check for the balance -0 logout .

Login as A – apply for OD for the same amount 20,000- click apply and log out.

Login as Manager – approve the OD of A –logout

Login as A check the balance it should be 20,000-n logout

Change the server timing for the next 30 days

Login as A and check the OD balance 20,400- click on re pay pay20,400 and check the OD balance it should be 0.

Development environment

It is the set up which is used for developing the s/w.

It contains h/w,s/w, n/w and server.

Test environment

It is the setup which is used for testing the s/w, It contains h/w,s/w, n/w and server.

Production environment

It is a setup which is used for running the s/w for the business, It contains h/w,s/w, n/w and server.

Note:

Whenever the new build comes in testing team concentrates on testing the new features first – because the probability of finding the bugs id more, we expect more no. of bugs in the new feature –

As soon as new build comes in,

Test the new features

Do integration testing

Retest all the fixed defects / bugs

Test unchanged features to make sure that it is not broken

Each test engineer retests only his bugs which are fixed, he is not responsible for other bugs found by other test engineers.

We find new bugs in old features why?

Fixing the bug may introduce new bugs.

Adding new features (modules) might introduce the new bug.

We might miss the bug in the previous test cycle.

When we start system testing?

When all the basic features are stable

When minimum bunch of features are ready

Environment similar to production must be available.

When do you release the product to the production?

When all the features requested by the CM are ready

When all functionality, integration and the end to end business scenarios looking fine.

When there are no critical or blocker bugs left out in the product

We have to test the product in an environment similar to the production environment

Total pending bugs are very less and all are very minor and less than the acceptable limit by CM

The testing the environment should be similar to the production environment means,

The h/w should be similar to the production

The make should be similar to production server (ex.. if the production server is HP, then the test server should also be HP server)

Configuration and make must be similar, but with different capacity that is no. CPU

The s/w should be similar to production

The OS should be similar

Application server should be similar

Web server should be similar

Data base should be similar

In testing environment, who is involved in installing the software?

Test engineer (anybody from testing team)

Anybody from development team

Release engineer / Build engineer

Release

The entire time period right from collecting the requirement to delivering the s/w to the client or CM is known as Release.

Test Cycle

One Test cycle is the time or the effort spent to start testing and to complete (finish) testing the complete application or s/w.

One test cycle might take 3 days or 7 days or 5days it depends on

Size of the application

Complexity of the application

No. of test engineers available for testing

Re-spin

Getting one more build between one cycle is called respin

Build

All the programmers we compile we get binaries and all the binaries compressed we get compress, this compressed file is called Build

Patch

Patch contains modified code, added program and deleted program.

When we start the work we may get a chance to work on three types of s/w or Application

Stand alone application: Stand alone application are installed in the same system and test.

Ex. Notepad, paint, auto cad …..

Clint/server application: Clint is a graphical user interface provided to user

Server authenticates the user and stores the data

In client server application we have 2 s/w

Clint s/w

Server s/w

Ex. Face book app installed in mobile phone or in the system

Web application: any application access through browser is called web application. Web application is a kind of client application where in browser acts behaves like a client.

Ex. Face book application launched in the browser

Compatibility test:

Testing the s/w with in multiple environments is called compatibility testing. With different s/w or H/w environments.

it is done only for client s/w.

Performance testing:

Testing the stability and response time of an application by applying load is called performance testing.

Stability: it is an ability to withstand desired no. of users load (load means no. of users)

Response time: T1+T2+T3

T1 => time to send the request

T2 => time to run the program in server

T3 => time to response

T1&T3 depends on Internet speed or the network speed, so we should have good network speed.

T2 depends on server.

How to do performance testing

There are many tools available more famous one is load runner and J-meter.

Types of performance testing

Load testing: Testing the stability and response time of an application by applying a load which is equal to or less than desired no. of users is called the Load testing. The desired no. users will be mentioned in the requirement specification.

Stress testing: Testing the stability and response time of an application by applying more than the desired no. of users is called stress testing.

Volume testing: Testing the stability and response time of an application by transferring huge volume of data through the application is called volume testing.

Soak testing: Testing the stability and response time of an application by applying load continuously for particular period of time is called soak testing.

For what kind of application we do performance testing?

For any application used by multiple users should do performance testing

Any application which generates revenue

For the features that used by more no. of users frequently we do performance testing.

When we do performance testing?

It varies from platform to platform for certain platform they start performance testing in designed stage itself.

In some project we think about the performance testing when the product is functionally stable.

Reliability testing

Testing the functionality of an application for a particular period of time.

Acceptance testing:

Testing done by the testing team in the customer place is called acceptance testing.

Why customer should do acceptance testing?

Chances are there that the s/w or the product is delivered with the lot of pressure about deadline to launch the product this cause lots of bug.

Chances are there that the developer and test engineer misunderstand the requirement and develop wrong feature to find that they do AT.

Under the business pressure s/w Company might push the s/ w with lot of critical bugs to avoid that they do acceptance testing.

With critical bug if the customer launch the product chances are there that they might undergo seviour loss, to avoid that they do AT

Approach 1:

It is a end to end testing done by engineers sitting at customers place where in they take all the real time business scenarios and check whether it is capable of handling it or not.

Approach 2:

It is an end to end testing done by end users where in they use the software for particular period of time and check whether it is capable of handling the real time business scenarios or not.

When customers have less no. of IT employs they launch the product for testing at the end user or else to the agent. They start using the s/w along with the old manual writing. If they got any problem they will contact the IT team and the IT physically go to the end user place and understand the problem.

And again they come and check the requirement that they given to the s/w company if the problem/ requirement is mention in the requirement they send it to the s/w company as a bug or else they prepare changes request.

This type of acceptance testing is called end user acceptance testing.

More no. of acceptance testing cycle means what?

Testing team is not done god job they might have missed many bugs that customer has catched.

The requirement which was given in the beginning is not cleared

After launching or giving the product to customer he is getting new ideas so he is asking for the lot of changes

Approach 3

It is an end to end testing done by our own TE sitting at CM place where in they refer user scenarios given by CM and check whether S/W is capable of handling the real time business scenarios.

Approach 4

It is an end to end testing done by our own TE sitting at our own company where in the refer CM’s user scenarios given by CM and check whether s/w is capable of handling the real time business scenarios

Hot Fix

The critical or blocker bug present in the production that could be fix is called HOT FIX.

Root Cause Analysis (RCA)

Write the defect and write the cause for the defect or reasons that why the defect came is called RCA or Fish Bone Technique.

Smoke Testing or Build verification testing or dry run or skim testing

Testing the basic or critical features before we do through testing is called smoke testing.

It is also called build verification testing because we check whether the build is broken or not.

NOTE:

Here we do only positive testing

Here we test only critical features we do not test all the features

Assignment

List out the features that are to be tested as part of smoke test for any 2 application

When we do smoke test?

Whenever the new build comes we should start with Smoke testing, because developers are giving new build means they would have done the changes (addition, modification, removal of features or bug fixes). This might effect basic or critical features so we should start with smoke testing.

Developer before giving the build to the testing team they do smoke testing

After installing the s/w in the production by release engineer or support engineer they should do smoke testing.

Whenever the product is given to the customer for Acceptance testing they should start with smoke testing

Every cycle of acceptance test customer should do smoke test.

Why we do smoke testing?

To check whether product is testable or not,

In the beginning only test all the critical features if you find too many bugs in the beginning itself it means the product is not eligible for further testing, so we can stop testing and spend all the remaining time in identifying the so other scenarios.

If we find bug in critical features in the beginning itself you can send it to developer so that developers have sufficient time to fix.

Doing smoke testing there we can confirm that the product is installed properly

It is a kind of health check of the software

We do this to check whether we have received broken build from developer

How do we manage smoke testing?

In some company we manage it manually (writing the test cases and test)

In some company we manage it automatically (we run automation script)

We have written all the smoke test cases for all the critical features, whenever new build comes we execute it in the beginning.

**AD-HOC testing also called Monkey testing / Gorilla testing:**

Testing the application randomly is called Adhoc testing.

Here we don’t refer formal documents like test case to test script.

Why we do Adhoc testing?

When we do Adhoc testing?

Questions

1. Difference between acceptance testing and system testing?
2. What is acceptance testing?

Definition, and tell why we do

1. Why we do acceptance testing?

Tell all the points

1. What is smoke test?

Tell what - definition

Tell why we do smoke testing (tell all the points)

Tell when we do

1. Why we do smoke testing?
2. When we do smoke testing?
3. What is the difference between smoke and sanity?
4. How do you manage smoke testing in real time project?
5. What is Adhoc testing?

Tell what; tell why we do, when we do

1. Why we do Adhoc testing?
2. When we do Adhoc testing?
3. Can you explain some examples how you do Adhoc testing?

You can explain the examples given in the class

1. What is hot fix?
2. What is release?
3. What is test cycle?
4. What is respin?
5. What is patch?
6. What is RCA?
7. What is build?
8. What is bug?
9. What is defect?
10. What is failure?
11. What is error?
12. What is positive testing?
13. What is negative testing?
14. What is Functional testing?

Tell the what, and tell about positive testing and negative testing

1. What is integration testing?

Tell definition, tell all types of integration testing with definition,

how to do IT,

when to do?

1. When we can start the integration testing?
2. How to do integration testing?
3. What are stubs and drivers?
4. What System testing?
5. When we can start the system testing?
6. When do you release the product to production?
7. What are Development environment / test environment / production environment?
8. What is performance testing?

Tell what and tell types with definition

Project specific

1. How many cycles of acceptance testing your customer have done?

Tell anywhere between 3-8 cycles

1. Did they find any defects while doing acceptance testing?

Yes, there were few minor defects.

1. How you have done the Adhoc testing in your current project?

Take some creative examples and explain

1. What type of application you have tested?

Web app /client server

1. How to do performance testing using tool?

Tell about tools

**Exploratory testing**:

Explore the application, understand the application, based on understanding identify all possible scenarios, document the scenarios then test the product based on the documented scenario this is called as exploratory testing.

Draw backs of exploratory testing

Chances are there that we might miss understand the feature as bug

TE while testing he enter alphabet and he assume that username should accept only alphabet and when he enter alphanumeric it will accept but TE expecting error so he will send it as a bug to developer.

Chances are there that we might misunderstand the bug as feature

Ex. For phone no. text box customer gives the requirements for +91 and 10 digit, but while testing TE assume only 10 digit, he enter 10 digit and test, it should through error message but it is accepting so he miss this bug and he thought that this is a feature (according to the customer it is a bug)

Some time features we never come to know that why it is missing

While building 2nd build changes his requirement for delete the feature so developer deleted the module

When TE got the build 2 he came to know that some module is deleted and thought that it’s a blocker bug or critical bug.

**To overcome these drawbacks solutions are as follows**

Always interact with testing team, development team, business analyst or with the customer and try to understand the product.

Based on the domain knowledge try to test the product

By using the common scene we can test some text fields.

By referring and comparing similar applications which is already released in the market, try to test the product.

How do you manage the exploratory testing?

We manage ET by documenting the scenarios nothing but writing the test cases.

Why/ when we do ET?

When there is no requirement in the project

Requirement is there but there is no time to go through

Requirement is there but it is not understandable.

**Interview Questions:**

1. What is ET?

* Tell definition, why we do, drawbacks, how to overcome drawbacks

1. In my project I don’t have requirement how do you test the product or what is your approach?

* See, as a TE I can’t say don’t test the product when there is no requirement in the project, even though there is no requirement in the project I can test the product by applying exploratory testing technique.

What I will do is I will explore the application, understand the application, based on understanding I will identify all possible scenarios for FT, IT, ST and document the scenarios, then I will test the product based on documented scenarios.

While doing this type of testing on product chances are there we do some mistakes.

1. Tell the drawbacks of mistakes

2.

3.

For these drawbacks what is my solution is

Tell the solutions for ET

3. In my project I have less time what is your approach or procedure or how do you test the product?

=> see when time is very less our objective is

1. To find more and more bugs
2. Cover maximum possible features
3. At least make sure that important or critical features are tested
4. Avoid certain tasks
5. To find more and more bugs

* I will test new or modified features first so that I can catch more no. of bugs
* Whichever the features got impact I will test that features first so that I can catch more no. of bugs.
* I will test the feature which are developed by the new developer

By adopting Azile methodology we easily come to know that who joined newly and which feature he developed.

* I will test features which developed by developer who normally does mistakes
* I will test the feature which are developed by the new technology

1. Cover maximum possible features

To cover maximum possible features with less time what I will do is

* I will execute those scenarios which gives more coverage with less time
* I will avoid those scenarios which gives less coverage.

1. At least make sure that important or critical features are tested

To make sure that important or critical features are tested with less time what I will do is

* I will execute those scenarios which touches the maximum features

1. Avoid certain tasks

When I say avoid certain task means

* I will reduces effort spent on negative testing
* I will avoid Adhoc testing
* I will avoid Usability testing
* I will not do compatibility testing over all platform instead I will test the product on core or base platform

I also stretch myself to work for more hours and work on holidays and try to complete the task.

**Compatibility test:**

Testing the s/w with in multiple environments is called compatibility testing. With different s/w or H/w environments.

**Q. Why we do compatibility testing?**

1. A developer develops the application in one platform and TE test the application in similar platform, when it is launched to market end user might be using the application in some other platform and many of the critical features may not work, because of that bad name spreads and no. of customers who buy the product will reduce to avoid this we should do compatibility testing.
2. To check weather product works consistently in the all platform or not we do compatibility testing.
3. Developer would have written lot of platform specific code so in order to test it we should test it in multiple platforms.

Q. **When we do compatibility testing?**

=> When the product is stable in base platform will do compatibility test.

Base platform means which is used by more no. of users.

**Q. How do we choose the platform?**

Customer should give the system requirement specification in that they should mention in which platform the s/w should work (they will give the list of platform to be tested)

If the developer writes the platform specific code then it is not necessary to test it in all the platforms.

**Q. What kind of bugs we find while doing compatibility testing?**

=>

1. Lot of alignment issues

2. Object overlapping

3. scattered content

4. images with certain format may not be displayed in certain browsers.

5. scrollbar issues, some time horizontal or vertical scroll bar may not be displayed or it might be displayed but it may not work.

6. change in look and feel

7. some change in font size, colour and style.

**Compatibility issue:** features that are working in some platform but not working in some other platform

**Functionality issue:** features that are not working in all the platform then it is called functionality issue.

**Hardware compatibility test**:

Test on different processors – we test for make – Intel, ADM processor and also test for speed – 3.1GHz, 2.7GHz

Test on different RAM – we test for make – Sansung, Trancient and also we test for size 1GB, 2GB

Test on different motherboard – we test for diff make

Test on different VGA card (video Graphics Array) – Diff make

Hardware compatibility test can be done for standalone application.

Hardware compatibility test is not done for Web application.

**Globalization testing**

**Globalization:** Developing a software for multiple languages is called globalization.

Testing the application which is developed for multiple languages is called Globalization testing.

There are 2 types

**Internationalization testing (I18N testing)**

Suppose if we want the application in Chinese, then we click on the browsr it will take to the server where the program is in English, from there it is executed and the output is translated into Chinese and displayed in Chinese language.

Q. How do we do this translation of language?

By using a translation? NO. Then how do we do it ?

Drawbacks in using a translator

Meaning are changed

Not conveying the feeling. Thus we rolled out the usage of translato

**Localization testing (L10N testing)**

Format testing is nothing but Localization testing

or

Testing done for format specification according to region/country is called L10N testing

Let us see the different format testing we do in L10N testing

Currency format testing

Date format: DD-MM-YYYY

MM-DD-YYYY

Pin code format testing

Image format testing

**As a test engineer we should test 3 things**

* Entire s/w displayed in right language and every feature name, heading is displaying in rite language.
* All feature displayed in right place
* Entire s/w functionality working or not.

**Acceptance testing**

1. What is acceptance testing?

Tell what , why and explain the 4 approaches

Q. What are the types of Acceptance testing?

1. **Alpha testing** : It is type of testing done before launching product to the market, it is generally done by TE’s
2. **Beta testing:** It is a type of testing which is done after releasing product to the market, it is done by the TE’s and the end users.

Q. How many cycles of system testing you have done in your current project?

**90% of test cycle we have done the system testing.**

**Test Cases**

**Q. What are the draw backs to be seen in testing the product over the period of time?**

=>

1. There will be no consistency in test engineers.
2. Quality of testing depends on memory power of TE
3. Quality of testing varies from person to person
4. quality of testing depends on mood of the TE

**Test Cases**

Test case is document which contains all the possible test scenarios for a specific requirement.

It contains different sections like

Step NO., Description, Input, Expected result, Actual result and status.

**Q. When the test Engineers write test cases?**

=>

1. When Developer are building the product TE write the test cases.
2. When developers are adding the new feature TE will add the test cases for new feature.
3. When developers are modifying the features TE will modify the TC

Whenever the developer gives the build to testing team, they will start executing the test scenarios.

When developers are developing the s/w TE have sufficient time and at that time they should refer the story card and write all possible scenarios. When the build comes they can execute the scenarios and because of this they will not miss the scenarios.

**Q. Why do we write test cases?**

=> 1. To have better coverage

2. To have consistency in test execution that mean if you have documented all the scenarios you can make sure that you are executing it in all the test cases.

3. To depend on the process rather than on a person.

4. To avoid training every new engineer on the product or on the requirement.

5. Test case is the only document which acts like proof for our customer, developer and management that you have covered all the possible scenarios.

6. Test case acts like a base document for writing automation scripts, if you refer the test case and then write the automation scripts you can ensure that same kind of coverage is maintained even in automation script.

7. If you documented the scenarios you don’t have to remember the scenarios

8. If you have documented the test cases, test case execution happens in a very organized manner

9. Test execution time reduces.

**Test case format:**

Test case has 3 stages

* Header
* Body

|  |
| --- |
| Test case Name: Amount text field  Requirement NO.: 70  Module Name: Amount transfer  Pre Condition: Create an account with the name abc and xyz  Test data: abc :- username // Password, xyz :- username // password  Severity:High  Test case type:  Brief Description: test the amount transfer text field by entering all the possible scenarios. |

* Footer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl No.** | **Action/Description** | **Input** | **Expected Result** | **Actual result** | **Status** |
| 1. | 1. Enter the positive number in the amount text filed and click on transfer button. | 200 | Amount transferred message window should be displayed | Amount transfer message is displayed | passed |
| 2. | Enter the negative integer in the amount text field and click on transfer button. | -200 | An appropriate error message should be displayed |  |  |

|  |
| --- |
| Author: Thejaswi  Reviewed By: Nagesh  Approved By: Savith  Approved Date: 24-12-2021 |

**Test case Design technique:**

1. **Error guessing technique**

Guess all the possible errors or defects and derive the scenarios, we guess the error based on experience and intuition.

Ex:Req. Amount text between 100 to 5000

Error guessing

-100

100.5

12\_00

Rs.500

$500

Hundred

Blank

0

0500

5,000

1. **Equivalence partitioning**

* Pressmen

If the input is the range of values then design the test cases for one valid and 2 invalid value

Ex. Req for 100-5000

500 (valid)

6000 (invalid)

90 (invalid)

If the input is set of values then test the text field by entering one valid and 2 invalid value.

100 200 500 2000

If the input is Boolean then design the TC for both true and false.

Check male

female

* Practice

Divide the range into equivalent parts and test the application for all the values, make sure that you are testing the application for 2 invalid values.

If(amount 100-2000)

{

Deduct 2%

transfer

}

Else if(amount 2000-5000)

{

Deduct 3%

transfer

}

99

3000

1000

5001

1. **Boundary value analysis (BVA):**

If the input is the range of values between A and B then design the test case for

100

100-5000

, 100, 101

4999, 5000,

Ex: if the range of value is in between 100 and 5000 then we have to test it for the following inputs

99, 100, 101

4999, 5000, 5001

When comparing the equivalence partitioning and BVA, testing values are repeated – if that is the case, we can neglect Equivalence Partitioning and perform only BVA as it covers all the values.

**Interview Questions:**

1. What are the test case design techniques?

Q. How do you make sure that they review test cases seriously or completely?

🡺 . TL should pick some test cases randomly and review

🡺 Assign primary and secondary owners so that primary has seriousness to review.

**Review Ethics**

1. Always review the content not the author
2. Even after review if there any mistakes both author and reviewer are responsible.
3. While reviewing spend time only in identifying the mistakes not in identifying the solution for it.

Q. Why we review the test cases?

Or

If I give you few test cases what could be your approach to review?

Or

How do you review the test case?

1. First I will see the requirement for which the TC is written then I will go to the body of the TC and try to find

Missing scenarios

Repeated scenarios

Wrong scenarios

1. Check whether the scenarios are organized properly or not so that it should take less time for execution.
2. Check whether it is simple to understand so the new engineer will be able to execute it without asking the question
3. I will look into the header of TC and check weather all the attributes are covered or not.

I will check weather all the attributes are having relevant content or not

1. I will check whether TC template is according to standard, defined in the project.

**Procedure to write the TC**

1. System study: read the requirement and try to understand the requirement if you have any quarries interact with customer, developer and BA
2. Identify all the possible scenarios: Identify, Brain Storming Session (BSS), measure the BSS.

In BSS, presentation

Features

Improves product knowledge

Scenarios

Missing

Wrong

Repeated

1. Write the test case:

Document the identified scenarios

Group all related scenarios

Prioritize with the group

Apply test case design technique

Use standard test case format

Document

1. Review the test case
2. Fix the review comments
3. Verify the fix
4. Test case Approval
5. Store it in a repository

**Test case Repository (TCR):**

Q**. Where do you store the test case?**

=>

1. In Exl sheets

2. Word file

3. Shared file

4. Test management tool eg QC (Quality control), test link. Jira

5. Version control tool

**Q. How do you make sure that your test coverage is good?**

**Or**

**Q. How do you convince your customer or management that you have covered everything in testing?**

=> Our test coverage is good because our test cases are good

1. We have followed a strict procedure to write the test cases.

* I did through system study because of that I was able to identify more scenarios.
* I identified scenarios and perform Brain Storming Session, because of that I was able to find many missing scenarios.
* While writing the test cases I applied test case design technique because of that I was able to add more scenarios.
* We review the TC’s and found many missing scenarios because of that I was able to improve my test coverage.
* While executing the TC’s I found many new scenarios and I added them back into the test cases.
* I also perform Adhoc testing to improve the test coverage.

1. We prepare traceability matrix and ensure that every requirement got at least one test case.
2. We adopted or applied code coverage analysis technique and found that every source code which is not covered as a part of test execution, later on written the scenarios for that.

**Traceability Matrix:**

It is a document prepare to make sure that every requirement got at least one test cases.

Advantages of traceability matrix:

1. It gives traceability from high level to automation script.
2. We can ensure that every requirement as got at least one test cases, which intern gives confidence that we are testing each and every features at least once.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Module Name** | **High level Name** | **Detail Requirement** | **Test Case Name** | **Automation script Name** |
| 1. | Amount transfer | 1.1 FAN | 1.1.1 Should accept only the 10 digit No. | CBO\_AT\_FAN1 | AUTO\_CBO\_AT\_FAN1 |
| 1.1.2 Should accept only those No. which is created by manager. | CBO\_AT\_FAN2 | AUTO\_CBO\_AT\_FAN2 |
| 2. | Loan | 2.1 Personal loan | 2.1.1  2.1.2  2.1.3 | ------ | Manual test |
| 2.2 Home loan | 2.2.1  2.2.2  2.2.3 | ---------------- | ------------------  -------------------- |
| 2.3 Vehicle loan | 2.3.1  2.3.2 | --------- |  |

Q. What is the difference between test case review and traceability matrix.

|  |  |
| --- | --- |
| **Test case Review** | **Traceability matrix** |
| 1. Here we check weather all the possible scenarios are covered on a specific requirement or not. | 1. Here we check weather every requirement has got at least one test case or not. |
| 1. Here we doesn’t check every requirement got at least one test case. | 1. Here we doesn’t check weather all the possible scenarios are covered. |

**Test Management Tool:**

It’s a software used to record the test cases or the requirement and to write the test scenarios

**Usability testing:**

Testing the user friendliness of an application we call it as usability testing.

1. Look and feel should be good
2. Easy to understand
3. Easy to navigate
4. Easy to use
5. Small sentence should be used
6. Simple words should be used
7. It should take very less time to reach what users want
8. Within 3 clicks user should get what they want
9. Upper case letter should not be used
10. Lower case letters should not be used

Regression Testing:

Testing the unchanged features to make sure that it is no broken because of changes is called regressing testing. Here the changes can be addition, modification, removal of bug fixes.

Types of regression testing

1. Unit regression testing:

Testing only the bug fixed or changes made is called unit regression testing.

1. Regional regression testing

Test the changes and only the impacted region is called regional regression testing.

Q. How will you identified the impacted areas? or How will you do Impact analysis?

1. Based on product knowledge as a TE I will be know how each and every module works in depth and also I will be aware of how all the modules are related based on that knowledge I will be able to find the impacted regions.

Eg. If we changes attach file feature It will make an impact on compose mail, all mail, inbox, outbox, sent items, spam.

Eg. If I modify the login feature it will affect the features like remember p/w, forgot p/w.

1. By preparing the impact analysis matrix we will identify the impacted areas.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Changed feature | login | Rememper p/w | Forgot p/w | Compose mail | inbox | Out box | Sent item | spam | All mails |  |
| Attachment | No | No | No | Yes | Yes | Yes | Yes | Yes | Yes |
| Forgot password | Yes | Yes | Yes | No | NO | NO | NO | NO | NO |

1. To identify the impacted areas the TE will do stand up meeting there they will discuss the bug fixed and what are all the impacted areas.
2. By conducting the impact analysis meeting where in entire testing team meet and discuss about list of changes (addition, modification, bug fixes) and impacted area.

We should do this whenever new build comes before we start testing.

Disadvantages of Regional regression test is that we might missed some impacted areas and miss bugs. So in order to overcome this we do full regression testing.

Doing full regression testing will consume more time so, we use to do the full regression testing in between some cycles or we do it at every 5 or 10 cycles.

1. **Full regression testing**

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

Q. Why we do full regression testing and when we do?

=> 1. Whenever we have too many changes are made we go for full regression testing.

2. Whenever the changes are made in the core features or root of the product.

3. Last few test cycles we must do full regression testing since we are launching the product, at the end movement we do not like to get the issues.

Q. What are the disadvantages of manual regression testing?

1. It is repetitive in nature over the period of time it becomes monotonous.

2. Manually testing the application consume more time, because of that test cycle duration increases which in turn increases the total time taken to deliver the product to the customer.

3. Man power is more expensive, over the period of time test engineer’s may not be effective because regression testing becomes repetitive.

**Software Test Life Cycle (STLC)**

STLC is a part of SDLC.

STLC is a procedure to test the software and it has different stages

|  |  |
| --- | --- |
| SDLC | STLC |
| 1. It is a procedure to develop the s/w 2. It has different stages like requirement collection, analysis, design, coding and testing, installation and maintainance. | 1. I t is the procedure to test the s/w 2. It has different stages like system study, prepare test plan, and write test cases, traceability matrix, test execution. |

System study

Prepare test plan

Write test cases

Prepare traceability matrix

Test execution

Defect tracking

Test execution report

Or

Test summary report

Retrospect meeting

Or

Project close meeting

Or

Postmortem meeting

1. System study

Here we read the requirement and try to understand the requirement if we have any quarries interact with developer or business analyst or customer and get it clarified.

1. Test plan

Test plan document which drives all the future testing activities.

This is where we decide

* How many engineers are needed
* What each engineers should do in each stages
* How we should install the product and set up the test environment
* This is where we decide what are the types of testing we should conduct on the product.
* What should be the testing approach
* This is where we decide what are the features to be tested and features not to be tested
* Schedule for every activity

1. Write test case

Once the planning job is done we start writing the test cases. When I say write test cases this itself involves different stages like

Identify all possible scenarios

Write test cases

Review test cases

Fix the review comments

Verify the test cases

Test case approval store it in test case repository

1. Prepare traceability matrix

Once the test cases are ready the biggest question is what is the guaranty that every requirement have got at least one test cases, to ensure that we prepare traceability matrix.

1. Test execution

Once we are sure that every requirement has got at least one test case we start our execution.

Test execution is a stage where in we test the product for more than 40-60 times this is where we conduct all the types of testing and this is where all the test engineer are productive for the organization and this is where the test engineer spend maximum time .

1. Defect tracking

We are executing the test mean defiantly we catch defect every defect that we find that should be track in a very organized manner which is nothing but defect tracking.

1. Test execution report or test summary report

It is a document which contains how many test cases are there, how many test cases are executed, what is the percentage of passed test cases and what is the percentage of failed test cases.

We prepare the document at the end of every test cycles and send it to development team customer and manager.

This is the end of project from testing point of view; from customer point of view we have one more activity that is retrospect meeting.

1. Retrospect meeting or project close meeting or postmortem meeting

Here the entire testing team meet and discuss about List of activities or good activities and list of mistakes or wrong activities followed in the entire test life cycle.

We document it and the document is called retrospect document when next project or the same project or in a next sprint when we are preparing test plan we refer the old retrospect document and prepare the test plan in such a way that old mistakes are not repeated and old good activities are again adopted.

At the end 2nd release once again we do retrospect meeting you come to know that number of mistakes that we are done are very less

Now we will look in detail each step of the STLC.

## REQUIREMENTS COLLECTION / SYSTEM STUDY

The requirements can be in any of the following forms,

* + - CRS (Customer Requirement Specification)
    - SRS (System Requirement Specification)
    - FS (Functional Specification)
    - If we don‟t have requirements and if we are given only the application, then we do *exploratory testing*.
    - Use case

***Use Case***

**Use case** is a pictorial representation of requirements. It explains how the end user interacts with the application. It gives all possible ways of how the end user uses the application.

Below is shown an example of how a **use case** looks like,

**Pre-condition : ………….**

**Action : ………….**

**Post-condition : ……………**

The above figure shows a sample **use case** of one of the requirements in the CRS. For the module A of the application, there are 7features.



**User 32 : USER PRIVILEGE**

**FULL ACCESS**

**ADMIN**

**A**

**Module**

**PARTIAL ACCESS**

**PAID USER**

**NO ACCESS**

**FREE USER**

**FULL ACCESS :-**

**……….**

**……….**

**……….**

**……….**

**……….**

**……….**

**……….**

**7 FEATURES**

**PARTIAL ACCESS :-**

**……….**

**……….**

**……….**

**……….**

**4 FEATURES**

Admin has access to all the 7 features. For a paid user – access to 4 features

For a free user – no access to any of the features.

**Ex** – for admin

*Precondition* – admin must be created

*Action –* login as paid user

*Post condition* – 4 features must be there

**Ex –** for free user

*Precondition –* free user must be created

*Action –* login as free user

*Post condition* – no features

**Who writes use cases**

**CRS**

**CUSTOMER**

**Development Team**

**Use Case**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Review Use Case** | | **CUSTOMER** |
|  | | **APPROVAL** | |

**TESTING TEAM DEVELOPMENT TEAM**

**Test Plan**

**Design**

**Test Case**

**Coding**

Customer gives the CRS for the application to be developed. The development team write the **use case** for the CRS and the **use case** is sent to the customer for review. If the customer approves it, then the approved **use case** is sent to the development team for design and coding. The approved **use case** is also sent to the testing team who start writing test plan and later on start writing test cases for the features of the application.

##### Difference between use case and prototype

**Use case** – talks about how the product should work. It is a pictorial representation of the application and its various features and also how they should work.

**Prototype** – here, we will not see how the end user interacts with the application. It‟s just a screenshot of the application (exact image of the application)

**How developers develop *use cases***

Developers use standard symbols to write **use cases** for universal understanding. He uses UML – Unified Modelling Language to develop **use cases**.

There are readymade tools to write use cases – like **Rational Rose**. It has readymade UML symbols – we can just drag and drop them to write **use cases** – developers use these symbols and write **use cases**.

**When you join a company, always ask for** –

* + - Requirements of the project
    - Test plan of the project
    - Test cases(existing) of the project
    - Application which is to be tested.

## TEST PLAN

Test plan is a document which drives all future testing activities.

Test plan is prepared by **Test manager(20%)**, **Test Engineer(20%)** and by **Test Lead(60%)**. There are **15 sections** in a test plan. We will look at each one of them below,

1. ***OBJECTIVE*** :- It gives the aim of preparing test plan i.e, why are we preparing this test plan.
2. ***SCOPE :-*** This covers features to be tested and features not to be tested
   1. **Features to be tested**

For ex, Compose mail Inbox

Sent Items Drafts

##### Features not to be tested

For ex, Help

…

…

…

…

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

**How do we decide this** (which features not to be tested) **?**

1. “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)

**D**

**D**

**C**

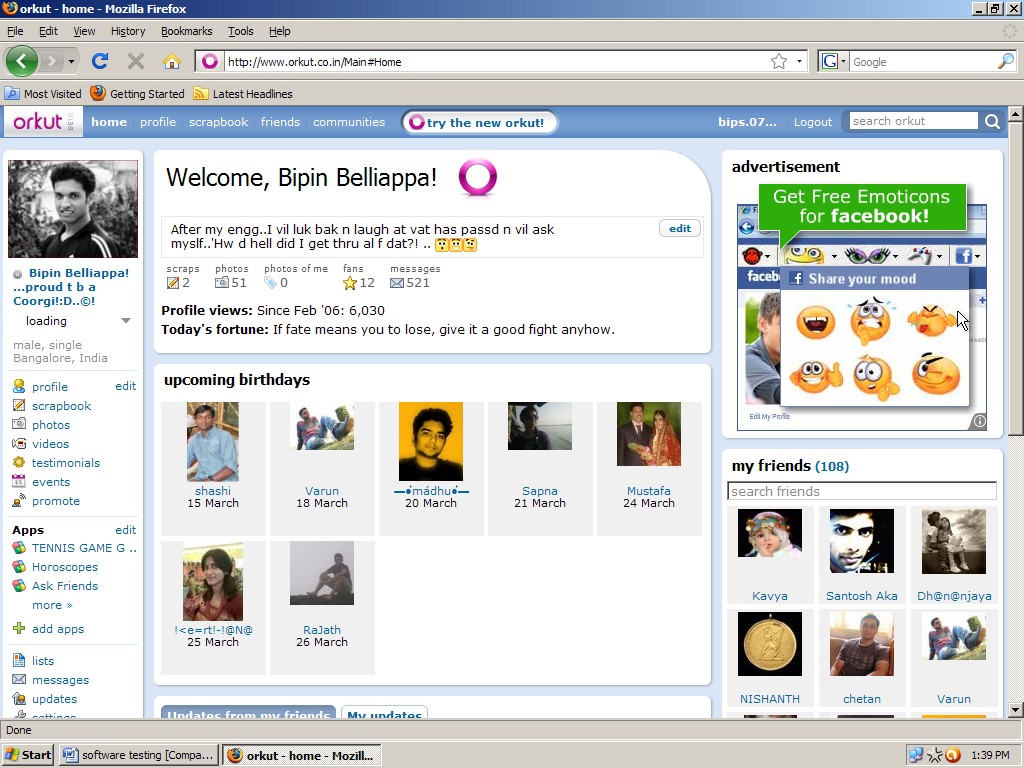
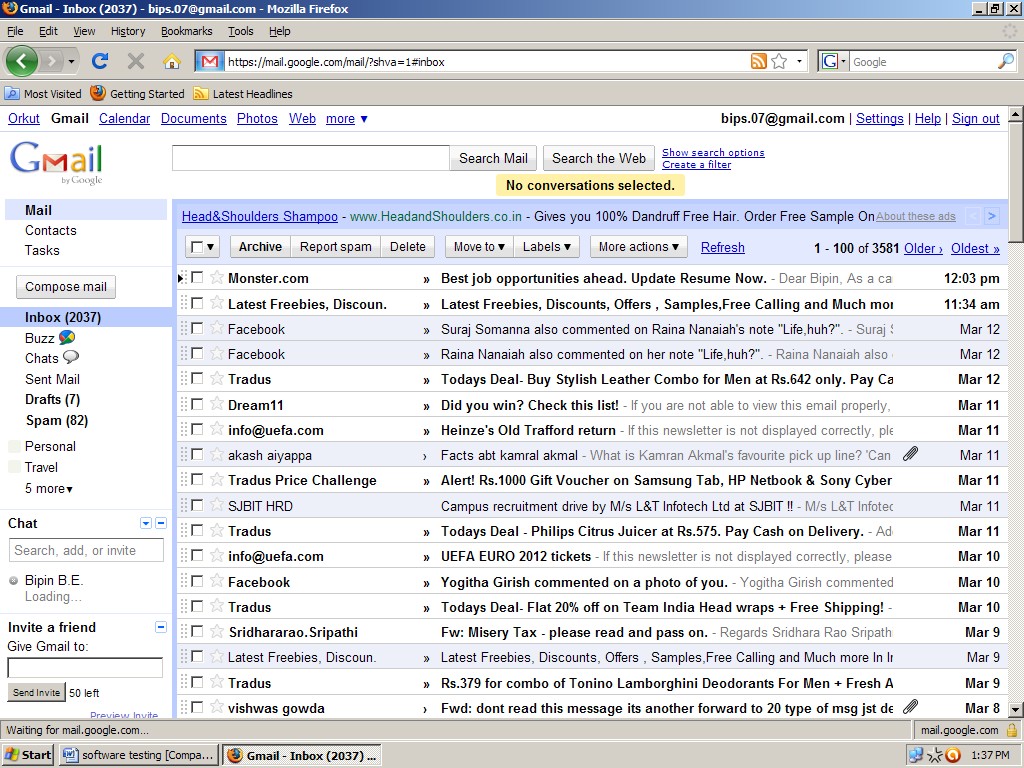
**B**

**A**

Let us consider that an application with features A, B, C and D are to be developed as per requirements. But then, D has already been developed and is in use by another company. So, the development team will purchase D from that company and integrate with the other features A, B and C.

Now, we will not do functional testing on D because D is already in use in the market. But we will do integration testing and system testing between A, B, C and D because the new features may not work with D properly.

##### 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.

Let us consider the **example** of Gmail. When we log into gmail, we see many links to other applications like

* FaceBook, LinkedIn, youtube, orkut etc. when we logged into Gmail and when we click on the orkut link – it must take us to Orkut‟s homepage.

Such features are called *Single sign-on feature* – it is a feature wherein 1 login allows access to multiple applications.

**d)** In the 1st release of the product – features that have been developed are – a, b, c, d, e, f, g, h, … m, n, o. Now, the customer gives requirements of new features to be built for enhancement of the product during the 2nd release. The features to be developed are – p, q, r, s, t.

During test plan, we write scope,

##### Scope

**Features to be tested**

P, Q, R, S, T (new features) A, B, C, D, E, F

##### Features not to be tested

G, H, I, J, … N, O

Thus we first test new features and then test old features which might be affected by building the new features i.e, impact areas. We do regression testing for A, B, C, … F.

###### TESTING METHODOLOGIES (Types of Testing)

Depending upon the application, we decide what type of testing we do for the various features of the application. We should also define and describe each type of testing we mention in the testing methodologies so that everybody (dev team, management, testing team) can understand, because testing terminologies are not universal.

**For example,** we have to test [www.shaadi.com](http://www.shaadi.com/), we do the following types of testing,

|  |  |  |
| --- | --- | --- |
| Smoke testing | Functional testing | Integration testing |
| System testing | Adhoc testing | Compatibility testing |
| Regression testing | Globalization testing | Accessibility testing |
| Usability testing | Performance testing |  |
| For standalone applications, like AutoCad, we do the following types of testing, | | |
| Smoke testing | Functional testing | Integration testing |
| System testing | Adhoc testing | Compatibility testing |
| Regression testing | Globalization testing | Accessibility testing |
| Usability testing | Reliability testing | Recovery testing |
| Installation / Uninstallation testing | | |

1. ***APPROACH***

The way we go about testing the product in future,

1. By writing high level scenarios
2. By writing flow graphs

###### By writing high level scenarios

for ex, we are testing [www.yahoo.com](http://www.yahoo.com/)

1. Login to Yahoo – send a mail and check whether it is in Sent Items page
2. Login to …….
3. …..

…..

…..

…..

This is written only to explain the approach to be taken to test the product. Only for the critical features, we will write a few very high level scenarios. We don‟t cover all scenarios here. That is the job of the respective Test Engineers for whom the features have been allocated.

1. ***By writing flow graphs***

##### Yahoo

**Finance Sports Mail Greetings**

##### Login

**Compose mail Sent Items Inbox Read mail**

##### Reply Reply all Forward

**Logout**

We write flow graphs because of the following advantages,

* 1. Merging is easy
  2. Coverage is easy

Flow graphs are written because writing high level scenarios is time consuming.

1. ***ASSUMPTIONS***

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

1. ***RISKS***

If the assumptions fail, risks are involved

1. ***CONTINGENCY PLAN OR MITIGATION PLAN OR BACK-UP PLAN***

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

Let us consider an **example for 5, 6, 7**

**B**

**A**

In the project, the **assumption** we have made is that all the 3 test engineers will be there till the completion of the project and each are assigned modules A, B, C respectively. The **risk** is one of the engineers may leave the project mid-way.

**C**

Thus, the **mitigation plan** would be to allocate a primary and secondary owner to each feature. Thus, one engineer quits – the secondary owner takes over that particular feature and helps the new engineer to understand their respective modules.

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

1. ***ROLES AND RESPONSIBILITIES***

##### Test Manager

##### Test Lead Test Lead Test Lead Test Lead

**Senior Test Engineer Junior Test Engineer Fresher**

When a Big project comes, it‟s the Test Manager who writes the test plan.

If there are 3small projects, then Test Manager allocates each project to each Test lead. The Test lead writes the test plan for the project which he is allocated.

##### Test Manager

* + - Writes or reviews test plan
    - Interacts with customer, development team and management
    - Sign off release note
    - Handle issues and escalations
    - ….
    - ….
    - ….

##### Test Lead

* + - Writes or reviews test plan
    - Interacts with development team and customers
    - Allocates work to test engineers and ensure that they are completing the work within the schedule
    - Consolidate reports sent by Test Engineers and communicate it to development team, customers(if it is a time&material project) and management
    - …
    - …
    - …

##### Test Engineer 1

* + - Review test plan
    - Write test cases for trend analysis
    - Asset survey
    - Write traceability matrix
    - Review test cases written for sales and purchase modules
    - Execute test cases written for trend analysis, asset survey, registration (old module developed in previous release. Adding trend analysis and asset survey has affected. Old module has been affected. So do regression testing)
    - Perform compatibility testing using Internet Explorer, Mozilla Firefox and Google Chrome in Windows XP and Windows Vista
    - Prepare test execution report and communicate it to Test lead.
    - ….
    - ….
    - …

##### Test Engineer 2

* + - Set up and install the product
    - Identify test cases to be automated
    - Automate identified test cases using QTP
    - Execute and maintain automation scripts
    - …
    - …

###### SCHEDULES :-

This section contains – when exactly each activity should start and end? Exact date should be mentioned and for every activity, date will be specified.

**Systen study write test cases execute test cases release date**

**Here we write test cases**

**00/00/00 00/00/0000 .. ….**

**B01 B02**

**ST**

**IT**

**FT**

Date 00/00/0000 Date 00/00/0000 Date 00/00/0000 Date 00/00/0000

This date is given by This date is given by customer development team

Thus, as we can see from the above figure – for every specified activity, there will be a starting date and closing date. For every build, there will be a specified date. For every type of testing for each build, there will be a specified date.

1. ***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. We should also mention the priority of the defect – high, medium, low.

##### Procedure to track the defect

….

….

….

….

##### Defect tracking tool

We mention the name of the tool we will be using to track the defects

##### Severity

* + 1. *Blocker*

….

…. (Define it with an example in the test plan)

For ex, there will be bug in the module. We cannot go and test the other modules because this blocker has blocked the other modules.

* + 1. *Critical*

…

… (define it with an example)

Bugs which affects the business is considered critical

* + 1. *Major*

…

… (define it with an example)

Bugs which affects look and feel of the application is considered as major

* + 1. *Minor*

…

… (define it with an example)

##### Priority

* + 1. *High – P1*

*…*

* + 1. *Medium – P2*

*…*

* + 1. *Low – P3*

*…*

*…*

*… P4*

So, depending on the priority of the defect(high, medium or low), we classify it as P1, P2, P3, P4.

1. ***Test Environment***

##### Hardware

* + 1. *Server* :- Sun Starcat 1500

(this is the name of the server from which testing team take the application for testing)

* + 1. *Client :-*

3 machines with following configurations, Processor : Intel 2GHz

RAM : 2GB

…

…

…

(this gives the configurations of the computers of the Test Engineers i.e, the testing team)

##### Software

* + 1. *Server*

OS : Linux

Web Server : TomCat Application Server : Websphere

Database Server : Oracle (or) MS – SQL Server

(the above servers are the servers which the testing team will be using to test the product)

* + 1. *Client*

OS : Windows XP, Vista, 7

Browsers : Internet Explorer, Internet Explorer 7, Internet Explorer 8, Mozilla FireFox,

Google Chrome

(the above gives the various platforms and browsers in which the testing team will test the product)

##### Procedure to install the software

…

…

…

(Development team gives how to install the software. If they have not yet given the procedure, then in the test plan, we just write it as TBD – to be decided)

1. ***Entry and Exit Criteria***

**EXIT 1)Based on %age test execution 2)Based on %age test pass**



**3) Based on severity**

**ENTRY**

**ST**

**IT**

**FT**

1. **WBT should be over …..**
2. **Test cases should be ready**
3. **Product should be installed with proper test environment**
4. **Test data should be ready**
5. **Resources should be available**

Before we start with Functional Testing, all the above entry criteria should be met.

After we are done with FT, before we start with Integration Testing, then the exit criteria of FT should be met. The percentage of exit criteria is decided by meeting with both development and test manager. They compromise and conclude the percentage. If the exit criteria of FT is not met, then we cannot move onto IT. Based on severity of defects means,

The testing team would have decided that in order to move onto the next stage, the following criteria should be met,

* + There should not be more than 20critical bugs
  + There should not be more than 60major bugs
  + There should not be more than 100minor bugs.

If all the above are met, then they move onto the next testing stage. But the problem with the above method was,

21 critical, 50major, 99minor – cant exit because there are more than 20 critical bugs. 10critical, 90major, 200 minor – can exit. But the 10 critical bugs can affect the product.

Thus, they came up with the concept of “*weight of defects*”. i.e, 3major = 1 critical, 5minor – 1critical and total critical should not be more than 60.

So, for,

21 critical – 21 50major – 16critical 99minor – 19critical

Totally there are 56critical bugs, so we can move onto the next stage. But for the 2nd example, we cannot move on.

##### 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 %age 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.

###### Note : All the numbers given above are just for example sake. They are not international standard numbers!!!.

1. ***Customer gets 100% defect free product means,***
   1. ***Testing team is not good***

***c) Product is old***

***INTERVIEW QUESTIONS***

***b) Developers are super***

***d) All of the above***

***Ans) a) is correct. Testing team is not good – because – fundamentals of software testing says there is no product which has zero defects.***

1. ***TEST AUTOMATION***

##### Features to be automated

**…**

##### …

**…**

##### Features not to be automated

**…**

##### …

**…**

##### Which is the automation tool you are planning to use

**…**

##### What is the automation framework you are planning to use

**…**

We automate the test cases only after the 1st release (*we have studied this earlier*).

* 1. On what basis do we decide which feature to be automated ?

Sent Items

Inbox

Compose Mail

Very Important Features

If the features are very important and need to be repeatedly tested, then we automate that feature. Because manually testing the feature takes longer time and also becomes tedious job.

* 1. How to decide which features are not to be automated ?

 For ex, “HELP” is a feature that is not repeatedly tested – so we don‟t have to automate it.

 If the feature is unstable and has lot of defects – we will not automate because it has to be tested repeatedly manually.

 If there is a feature that has to be repeatedly tested, but we are predicting a requirement change for that feature – so we don‟t automate it as changing the manual test case is easier than changing the automation script.

1. ***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,

* **14.1** Test Plan
* **14.2** Test Cases
* **14.3** Test Scripts
* **14.4** Traceability Matrix
* **14.5** Defect Report
* **14.6** Test Execution Report
* **14.7** Graphs and Metrics
* **14.8** Release Note

##### Graphs and Metrics

Here, we will just mention what are the types of graphs we will deliver and also give a sample of each graph we will be delivering.

##### (Defect Distribution Graph)

*Graph 1* :- in this graph we depict – how many bugs have been found and how many bugs have been fixed in each module.

**(Build-wise Graph)**

**(Defect Trend Analysis Graph)**

*Graph 2 :-* in this graph, we depict – how many critical, major and minor bugs have been found for each module and how many have been fixed for each module.

*Graph 3 :-* in this graph, we depict – build wise graph i.e, in each build how many bugs have been found and fixed for each module. According to the module, we have found defects. Adding C has introduced a lot of bugs in A and B. Adding D has introduced a lot of bugs in A, B and C.

*Graph 4 :-* Defect Trend Analysis graph depicts – this graph is prepared every month and we must send it to management. It‟s a kind of forecast. By the end of the project, “rate of fixing defects” curve must have an upward trend. Test Lead prepares this graph.

*Graph 5 :-* Test manager prepares this graph. This graph is prepared to understand the gap in estimation of defects and the actual defects that have occurred. This graph helps in better estimation of defects in the future.

###### Metrics

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | **Critical** | | **Major** | | **Minor** | |
| **Found** | **Fixed** | **Found** | **Fixed** | **Found** | **Fixed** |
| Sales | 40 | 36 | 80 | 30 | 90 | 15 |
| Purchase | .. | … | … | … | … | … |
| Asset Survey | … | … | … | … | … | … |

***Defect Distribution Metrics***

We generate the *defect distribution graph*(graph 1) by looking at the above data. Similarly we can generate many such metrics.

##### For ex,

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Engineer Name** | **Critical** | | **Major** | | **Minor** | |
| **Found** | **Fixed** | **Found** | **Fixed** | **Found** | **Fixed** |
| Bipin | 40 | 36 | 80 | 30 | 90 | 15 |
| Rajath | .. | … | … | … | … | … |
| Amith | .. | … | … | … | … | … |

In the above graph, we are maintain a record of all the test engineers in the project and how many bugs have been caught and fixed etc. We can use this data for future analysis. When a new requirement comes, we can decide who to give the complex feature for testing based on the number of bugs they have found. We will be in a better position to know who can handle the complex features very well and find maximum number of bugs.

***Interview Questions and Tips***

1. ***What is Metrics ?***

***Ans) We can tell any of the above.***

1. ***On the last day of the project i.e on release date, we find a critical bug. Then what will you do? Will you release the product or fix the critical bug?***

***Ans) 1st say – Testing team prepares a report and sometimes also provides a suggestion on what can be done. But, it‟s the Management team which takes a decision on whether to release the product or not.***

***But, now the Interviewer asks you – “I am asking what will you do, not what the management will do?” Then answer like this – “I will not release the product with critical bug because I want to deliver a high quality product”.***

##### Release Note

**Application developed. Released to the customer. Name of the release is “Cheetah Release”**

The product is developed and tested and released to the customer. The name of the release is “Cheetah Release”.

The release note contains,

1. *List of pending/open bugs*

…

…

1. *List of features added, modified or deleted*

…

…

1. *Platforms*(OS, Browsers, Hardware) *in which the product is tested*

*..*

*..*

1. *Platforms in which the product is not tested*

*…*

*…*

1. *List of bugs fixed in current release which were found in previous release production*

*..*

*…*

**TIGER RELEASE**

**CHEETAH RELEASE**

Let us consider that Cheetah release is the 2nd release of the product after the 1st release Tiger release. Some of the bugs found in the 1st release has been fixed in the 2nd release. Also a list of features which have been added, modified and deleted from the 1st release to the 2nd release will be mentioned here.

1. *Procedure to install the software*

*..*

*…*

1. *Version of the software*

*..*

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

1. ***TEMPLATES***

This section contains all the templates for the documents which will be used in the project. Only these templates will be used by all the test engineers in the project so as to provide uniformity to the entire 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
  + …
  + …
  + …

## This is how a Test Plan document looks like,

Pg 1

**CBO\_Testplan**

**Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Author** | **Reviewed By** | **Approved By** | **Comments** | **Approval Date** |
| 1 | .. | … | Name of manager | Version 1.0 is developed | dd/mm/yyyy |
| 1.1 | .. | .. | " " | Version 1.1 is developed. XYZ feature is added | dd/mm/yyyy |
| … | … | … | … | … | … |
| … | … | … | … | … | … |

Pg 2

1. Objective pg 1
2. Scope pg 2
3. Approach pg 3

..

…

…

…

##### TABLE OF CONTENTS

Pg 3 – Pg 19

….

….

…

….

….

….

….

….

….

….

**Entire Test Plan document**

Pg 20

**REFERENCES**

1. CRS
2. SRS
3. FS
4. Design document

…

…

…

In the 1st page – **we initially fill in only *version(write as Draft 1.0), author, comments and reviewed by.* Later on when the manager approves it, we fill in the *Approved by and approval date* and also remove (Draft) written in version column.**

Generally Test Engineers only review it and the test plan is approved by Test Manager.

When new features come, we modify the Test Plan. Change the version and whichever features need to be changed. Again it is reviewed, updated and approved by manager. Test Plan must be updated whenever changes happen.

References (Pg 20) contains all the documents that are used to write the test plan document.

###### Who writes Test Plan ?

 Test Lead – 60%

 Test Manager – 20%

 Test Engineer – 20%

Thus, we can see from above – in 60% of projects, Test plan is written by Test Lead and so on as shown above.

###### Who reviews Test Plan ?

 Test Engineer

 Test Lead

 Test Manager

 Customer

 Development team

Test Engineer looks at the Test plan from his module point of view. Test Manager looks at the Test plan from the customer point of view.

###### Who approves Test Plan ?

 Test Manager

 Customer

###### Who writes Test Cases ?

 Test Engineer

 Test Lead

###### Who reviews Test Cases ?

 Test Lead

 Test Engineer

 Development Team

 Customer

###### Who approves Test Cases ?

 Test Lead

 Test Manager

 Custom

Verification

1., It involves review walk through and inspection

2. it involves different activities like requirement review test plan review test case review design review and code review.

3.Here we check weather are we building right product

Validation

1. It involves actual testing
2. It involves different activities like FT, IT, ST AT Smoke testing Adhoc
3. Here we check whether are we building product right

Static testing

1. It is a verification process
2. it involves different activities like requirement review test plan review test case review design review and code review.
3. To do this we need not execute a program

Dynamic testing

1. Validation processs
2. It involves different activities like FT, IT, ST AT Smoke testing Adhoc
3. To do this we may have to execute the program

6 sigma process – it is 3.41 defect per million oppertinities

**Azile Methodology**

Q. What is Azile ? Explain Azile Methodology

🡺

* The Azile methodology is a way to manage a project by breaking it up into several phases.
* In an **Azile team** we have BA, Project Manager, and Product owner, Developer, Test Engineer and Scrum Master.
* In Azile Methodology every we do **scrum meeting,** here we discuss what are the things we did in the previous day, what are the obstacles’, impediments we have stock yesterday and what are the things we want to do today. All these will be discuss in the scrum meeting.
* We do this scrum meeting every morning or in the evening and it will be of 15 to 20 mins and scrum master will drive this meeting.
* The software development will start once we got the product backlog, this product backlog consists of several sprint backlog in a sprint backlog we have several story card, each story card gives the description about the feature.
* Here we select few features that can be developed within a story point of 3-5 weeks and should complete one business cycle and the customer should be able to use it.
* In some cases to complete one business cycle it require more features to develop and to develop all these features it takes more than one month, so in this case we divide the features and develop, half of the features will develop first and after that we send it to customer and customer will do acceptance testing and keep it with them and after that in the next sprint we develop the remaining features.

**Story point:** The estimate time taken to develop one story card is called story point

**Story card:** one unit of delivery,

**Scrum:** It is a methodology that allows a team to self organizes and makes changes quickly in accordance with azile principle.

**Scrum meeting/ Daily scrum/ stand up meeting:**

**Sprint meeting**: retrospect meeting

**Sprint planning:** Here we have planning for that particular sprint.

**Product backlog:** it is a the complete requirement of the project

**Sprint backlog:** it is the requirement that we selected for a sprint