1. **Explain me about Software Testing Life Cycle (STLC)?**

Software Testing Life Cycle will have below phases: Every tester in software testing will go through below phases.

* **Understanding Requirements / User Stories:** Understanding Use Stories (if you are in Agile model), OR understanding requirements by going through Software Requirement Specifications(if you are in Non-Agile model) which we generally call as SRS
* **Clarify your doubts**: Raise your doubts / questions after going through above user stories or requirements and get them clarified
* **Writing the Test Cases**: Here you guys start writing the test cases based on understandings from above user stories / requirements. We generally use test management tools to write test cases. We have so many tools available in the market to use like TFS which is from Microsoft and Test Link etc. Most of the tools are similar to use.
* **Review of Test Cases:** Once test cases are prepared they will be reviewed by internal testing team. We call it as peer review. I mean let’s say we have 4 testers in the project and one tester test cases will be reviewed by other tester and provide review comments. We also send these test cases to client for review but not all the time. It differs to client to client. If client request we have to send for review and get them approved
* **Execute Test Cases:** Once system is given for testing team to do testing we execute the test cases which are prepared. IF any test case gets failed we go ahead and raise defect in defect management tool for tracking.
* **Defect Logging:** While doing the test execution we raise defect when we see actual result is not matching with expected result. Once we raise defect it will be discussed in triage calls and make active OR reject based on discussion with stake holders.
* **Defect Re-testing:** Once defect is activated after discussion development team will start working on these all defects which are agreed as defect.
* **Automation**: Now a days automation has become mandatory in software testing industry as most of the projects are running in Agile. Once one release / sprint is completed we do automation of manual test cases which generally cover functional flow.

1. **What is black box testing?**

Black box testing is a kind of testing where tester performs the testing without knowing internal structure of code OR Architecture of the system. Here we perform testing by giving input and validate output is correct as per the input given. Let’s take below two examples to understand it better:

* **Example 1**: Let’s take a calculator where it allows to perform some arithmetic operations. Let’s say I would like to add two 2 numbers, so just input number say 20 then press “+” then enter 40 then press “=” we get output as 60. That’s it.  We don’t know internally what is the code written to do this OR we don’t know what is the logic that is written in the code OR Architecture of system how it is defined. So out input is providing two numbers and validating the output is correct or not in this example we have to validate whether out is 60 or not.
* **Example 2**: Let’s take an ATM machine where we withdraw some amount. Here we insert card input your pin and select Withdraw option from menu and enter amount. Let’s say we have given input as $1000. We just have to validate to check whether amount what is given as output from ATM machine is $1000 or not. Here also we don’t what is technology in which ATM is implemented, we don’t know Architecture of the ATM, we don’t know what is the logic written in the code.

So in black box testing we just provide input and check output without knowing below

* What is the logic that is written in code to perform specific action
* How is the architecture of system is defined
* What is the technology in which system is developed

1. **What are different techniques in this black box testing**

We have below techniques which are used in black box testing

* Boundary value analysis (BVA)
* Equivalence Partitioning
* Error Guessing

1. **Explain Boundary Value Analysis Technique?**

We use Boundary Value Analysis testing in black box testing. We generally use this technique to test if any field is taking range of data. As the name says it always checks with the boundaries of the range. It checks for both positive and negative cases.

Let’s take an example: I have an application in which we have to enter employ number in order to create new employ in the system. Let’s say requirement given for this filed is it should be minimum of 1 and maximum of 1000 as minimum and maximum range. For this example if we use boundary value analysis we have to test with by entering following numbers

**Valid / Positive cases:**

* Enter 1 as input(this is minimum value)
* Enter 2 as input(this is minimum+1)
* Enter 1000 as input(this is maximum value)
* Enter 999 as input(this is maximum-1)

**Invalid / Negative cases:**

* Enter Zero as input(this is minium-1)
* Enter 1001 as input(this is maximum+1)

So here we are always testing application with boundaries of the minimum & maximum of the range that is given as part of requirement for a specific field. That’s it how we use this technique for testing.

1. **What is Equivalence Partitioning?**

Equivalence Partitioning is another block box testing technique which is used. In case of Boundary Value Analysis we try to input data with boundaries of minimum and maximum

In equivalence Partitioning we basically have two classes one is valid and other tow are invalid.

**Example**: Let’s take same example which understood from above question for BVA and see how we input in case of Equivalence Partitioning I have an application in which we have to enter employ number in order to create new employ in the system. Let’s say requirement given for this filed is it should be minimum of 1 and maximum of 1000 as minimum and maximum range. For this example if

We test with below one valid and two invalid classes using Equivalence Partitioning

**One Valid Class**:

Enter any number between the range which is nothing but 1 to 1000 so you can enter any number like say 90

**Two In-Valid Classes:**

Enter any number above the range which is nothing but more than maximum value which is 1000 in this case. so you can enter any number like say 2000

Enter any number below the range which is nothing but less than minimum value which is 1 in this case. so you can enter any number below range like say -10

1. **What is white box testing?**

In block box testing we understand that we do testing by just giving input and validate output without knowing any internal structure of the code, logic what is written, not knowing the technology and also without know the technology as well. But when it comes to white box testing it is reverse to black box which means you guys have to look at the code and verify  each and every line of the code and conditions for both true and false outcomes. This testing makes sure whatever code is written is correct and validates at code level.

1. **What are the white box testing techniques available**

Below techniques

* Statement Coverage Testing
* Condition Coverage Testing
* Branch coverage also known as Decision coverage

1. **Difference between Verification & Validation**

**Verification**: Verification happens in every phase of SDLC to make sure what we are building is correct or not. It is continues process that is performed by individual resource whoever is responsible for every phase of SDLC. Verification is performed using below

* Reviews
* Walkthroughs
* Inspections

**Validation:**We do validation to find out whether we build system right or not. Typically validation is performed by testing team.

In simple term to say “Are we building system right are known as verification” & “Did we build system is known as Validation”

**9. What is Static Testing and Dynamic Testing**

We do static testing before actual system is ready for testing. It means that we do reviews walkthroughs & inspections. Like we do Test Plan review, Test Case review etc before system is actually implemented. Dynamic testing we do once system is implemented and given for testing team to testing.

In simple words to say static testing falls under “Verification” & dynamic testing falls on “Validation”.

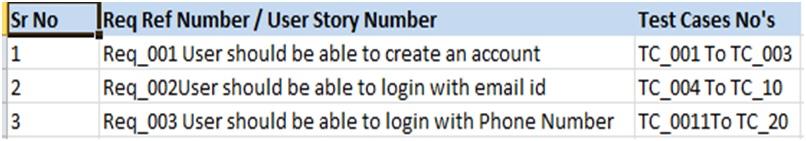
**10. Explain me about traceability Matrix**

Requirement Traceability Matrix or RTM captures all requirements OR user stories given by the client and it helps in tracing test cases which are written against these requirements.

In simple words to say, it is a document that maps and traces client requirement with test cases. The main purpose of Requirement Traceability Matrix is to make sure that testing team written test cases to cover client requirements so that no functionality will miss from testing team under testing.

**11. Can you write template for Traceability Matrix**

Let’s take an example we have requirements / use stories from client  show below, as a tester we write the test cases for every requirement / user story. In below Traceability Matrix template we can see every requirement / user story is mapped with Test Cases that are written by testing team. This way Traceability Matrix allows to make sure we cover all the requirements / user stories from testing team.



**12. Explain me different Test Levels we have**

We have below different test levels

* Unit Testing
* Integration Testing
* System Testing
* User Acceptance Testing (UAT)

**13. What is Integration Testing?**

Integration testing mean we do integration of each developer code and do this testing to make sure each developers code is working after integration of each developer’s code.

This testing is mainly performed to makes sure each component is interacts well without any issues so that we can prevent defects in system testing before it is giving to testing team..

**14. What is UAT testing?**

User Acceptance Testing is performed by client once testing team completed system testing. We get approval from client after doing this testing to make it available for end users.

**15. Explain difference between Test Scenario & Test Case**

**Test Scenario:**As a testing team we have to always understand requirements / user stories from client. So for every requirement we always write test scenarios considering what are the scenarios we have to test. For every requirement / user story we write one test scenario. Test scenario is mostly a one line statement which tells what we are going to test as part of this requirement / user story.

**Test Case:** Test cases will be in detail not like one statement when compared to test scenario. Test case will cover in detail like what are the steps to be covered, what data we have to provide & what is expected result for each step. We generally write multiple test cases for one test scenario.

**Example:**Let’s take an example to understand it better

Let’s say that we have below requirement / user story from client

“User should be allowed to register an account with FaceBook and duplication of user should not be allowed to create”

**For above requirement / user story we write below scenario and test cases**

**Test Scenario:** Verify user allowed to register an account

**Test Cases:** We write multiple test cases for the above scenario:

1. Create an account and check account is created
2. Check system is not allowed to create duplication of user
3. Verify GUI (Graphical User Interface) of Register Account page.
4. Verify filed level validation for Register Account page

**16. What are the different types of test cases you write?**

We always have to consider below types of test cases for each and every requirement / user story

* **GUI**: Graphical user interface test cases cover UI of application. Here we write test to check whether application screen showing as per the expectations which are given by client or not. Also we check look and feel of the screen like alignment issues, overlapping of the fields and spell check etc.. But for everything we need not to write a test case but we write at least one test case for checking UI.
* **Functional:**Here we write test cases to check the functionality of application is working as expected or not
* **Filed Level Validation:**Here we write test cases to check for each and every field for below
  + Mandatory Data Validation: If field is mandatory then leave that filed blank click on submit and see system throws valid error message
  + Max size data: Enter data more than maximum size specified. Let’s say FirstName filed supposed to take maximum of 100 letters then enter more than 100 letters and check proper error message is displayed or not.
  + Minimum size data: Enter data less than minimum size specified. Let’s say FirstName filed supposed to take minimum of 5 letters then enter less than 5 letters and check proper error message is displayed or not.
  + Invalid data: Let’s say FirstName field should take only letter then enter some invalid data like #%#%%%^1344 and check proper error message is displayed or not.

**17. Difference between Error, defect & Bug?**

**Error:** let’s say developer written code to implement addition of two numbers and where by mistake instead of “+” he added as “-“. So there is an error in the code instead of addition he is doing subtraction. This is called error in the code

**Defect:** Tester founds this error while doing the testing is called as a defect

**Bug:** Once defect is agreed as valid from development team then it is called as bug.

**18. Explain the Defect Life Cycle.**

Once tester finds a defect while doing the testing it will go with below life cycles

* When defect is found it will be raised in one of the defect management tool, it will be with status as “New” (status varies between the defect management tools)
* This defect will be taken with development team / stakeholders in “Triage” status to discuss on this defect
* If this defect is valid and agrees by development team then it will go to status as “Active” and will be assigned to developer. All the defects agreed as valid defects will go as Active.
* If this defect is not valid then it will go as “Rejected”
* Once developer starts working on it will go status as “In-Progress”
* Once developer fixes the defect it will go as “Resolved” and assigned back to tester for re-testing
* Once tester tests the defect and confirms it is working fine then tester will change status of defect to “Closed”
* If defect is still exist even after fix then tester will change status to “Re-Open” as it is not working
* Once defect is in “Re-Open” status by tester, it will go above process again like active, in-progress, “Resolved”

**19. What are the different status of defect?**

Status of defect varies from tool to tool but below are the common status in most of the tool

* New
* Triage
* Active
* In-Progress
* Resolved
* Re-Open
* Closed

**20. What is defect severity & priority?**

**Severity:** Tells how severely a defect is impacting to business / system

**Priority:** Tells urgency of fix of defect how soon it should be fixed

**21. Explain me different severities & priorities?**

Every defect management tool will have its own severities and priorities. But most the tools will have below

**Severities:**

* Critical
* High
* Medium
* Low

**Priorities:**

* P1
* P2
* P3
* P4

**22. As a tester you raised defect and developer is not accepting it as defect. What you do in this situation?**

It is common case in most of the projects as understanding of requirement differs from person to person. So have to explain with requirement reference number / user story number which you are referring as it is valid defect. We need to explain that we have not raised defect in assumptions and it is raised as it is not working as per the expected.

**23. What is Test Plan and explain me contents of Test Plan?**

Test Plan is a document which will have below contents

* What is this document about
* Assumptions
* In-Scope
* Out Of scope
* Entry criteria
* Exit criteria
* Risks
* Test Deliverables
* Automation and tolls planned to use

In Detail:

**Test Plan:** is a document which consistent of all the information related to testing like what are we going to test, in scope, out of scope, resource management etc..

Let’s assume we are working for banking application where we have two modules debit card and credit card. Consider we are working for debit card management and this how our test plan will go with respect to In-Scope and Out-of Scope

**In-Scope:**

Debit card management testing will be in scope of testing

Browsers & Operating Systems:

Testing will be performed in below browsers

IE with version IE8 and above

Chrome version with 42 above

Operating Systems: Testing will be performed in below OS

Windows with version 2008 and above

Android version 2.0 and above

IOS version with 3.0 and above

**Out of Scope:**

Credit management testing is out of scope

Browsers and Operating systems other than above is not in scope for testing

**Test Environment:** Test environment should be setup and access should be given testing team as well to Dev team.

**Types of Testing:**

1. Regression Testing: In this testing we consider end to end flow as part of regression. We planned to do this testing through automation only
2. Sanity Testing: We also perform sanity testing
3. Functional testing
4. Installation / Un-installation Testing (If applicable)

**Automation:**

We do functional automation and non-functional (performance testing) as part of automation

Selenium is the tool which will be used for functional automation

Load Runner is the tool which will be used for performance testing

We do in-sprint automation for every sprint.

**Entry Criteria:**

1. Test environment setup should be completed
2. Testing team provided access to this environment
3. All the test scenarios and test cases are prepared and ready for execution
4. Test cases are reviewed and approved by client (if applicable)
5. All the tools related to automation is purchased and ready to use
6. Test Management & Defect Management tools are purchased and ready to use
7. Team got all the relevant KT on domain
8. Team got training related to test management / defect management tools
9. All the test case are prepared and approved by client

**Exit Criteria:**

1. All the test cases are executed and 95% of the test cases are passed.
2. No test cases are in blocked status
3. All the defects with Critical, High & Medium severity defects are closed

**Risks:**

1. If there is a delay from development team of delivering to testing team will lead to slippage of committed release date to client. Mitigation plan will testing team needs to work on weekends to cover the delay.
2. Resource Management:
   1. What if resource becomes sick and will not be available for a week
   2. What if resource got some personal emergency and will not be available for a week
3. Too Many defects: If testing team raised too many defects and this would lead additional efforts for Dev & Testing teams. This will also lead to slippage of derivable times

**Deliverables**: Below are the deliverables

1. Test Plan
2. Test Cases
3. Test Summary Report
4. Automation Scripts

**24. When to stop Testing?**

When we met the Exit Criteria that is defined in Test Plan. In most the projects below will be exist criteria in common

* All the test cases are executed and no test case is not in run state
* 95% of the test cases are passed
* No test case is in blocked status
* No critical, high & medium defects are in open / active status

**25. What you do if no sufficient time is given for testing?**

* We plan to test core functionalities of the system
* We try to cover complete end to end flow of testing
* We may not concentrate on field level validation.

**26. What is difference between Test Metrics & Traceability Matrix**

**Test Metrics:** It is used get the progress of testing where do we stand on daily basis, weekly basis. Test Metrics helps in preparing reports like DSR(Daily Status Report) & WSR(Weekly Status Report)

**Traceability Matrix:** Is a document which helps to make sure we are not missing any requirement from testing point of view. Here we map the test cases against with requirements to make sure all the test cases are covered for given client requirements. In some projects we send this document to client as well.

**27. What is Regression testing**

This testing is performed to test to make sure newly added / modified code is not introducing any defects in existing system.

We do impact base analysis and consider only the test cases which are impacted as part new user story requirement.

We do regression testing in below scenarios:

* When new code is implemented as part of new sprint
* When huge defects fix is happened

**28. Can you explain me how you do impact base analysis for Regression testing**

Whenever we have to do regression testing, we need to consider what we have to test as part of regression testing. So for this we need to do impact base analysis. Let’s look at below example to understand in detail

**Example:**

**Sprint 1 / Release 1:** We got below user stories / requirements from the client and let’s say we have written 100 test cases for below 4 user stories / requirements.

1. User should be able to register an account
2. User should be able to add beneficiaries
3. User should be able to transfer funds within the bank
4. User should be able to raise a complaint through online once login

**Sprint 2 / Release 2:**We got below user stories / requirements from the client and let’s say we have written 50 test cases for below 4 user stories / requirements.

1. User should be able to raise cheque book request
2. User should be able transfer funds to other banks as well
3. User should be able to pay bills online after login
4. User should be able to apply for credit card

**As tester what we do here:**

**Step 1:** We do testing of all user stories / requirements as part of Sprint 1 / Release 1. Means we execute all 100 test cases as part of testing.

**Step 2:** Same as step 1 we execute 50 test cases which are written as part of Sprint 2 / Release 2

Now what’s next? Now it’s time to consider regression testing. So we need to do impact base analysis and consider what to test as part of regression testing here?

If you look at 4 requirements from Sprint 1 and 4 other requirements from Sprint 2.

Requirement 3 which we tested as part of Sprint 1 but as part of Sprint 2 if we see requirement 2 which says transfer funds to other banks.

In Sprint 1 we tested transfer funds with in the bank but here developer is modifying the code of transfer funds to implement funds transfer to other funds as well.

So in Sprint 1 we tested requirement 3 but in Sprint 2 requirement 2 has impact on sprint 1 story. So here we need to consider testing of user story 3 from Sprint 1 as part of regression testing.

**29. What is compatibility testing and what do you consider here as part of this testing**

This testing is performed to make sure system is compatible in different browsers, operating systems & devices. We need get requirement from client what exactly client is looking as part of compatibility of system. Means in which browsers, operating systems &devices system should be compatibility testing.

We need to consider below while doing compatibility testing:

**Browsers**: We reach client asking in which browsers application needs to be compatible. Let’s say client says application needs to be compatible in Chrome, IE & Firefox. Then we need to test application in all these browser. Also needs to check with client which version of the browser in which we needs to test

**Operating Systems**: We reach to client asking in which operating systems this application needs to be compatible. If client says this application needs to be compatible in IOS, Android& Windows. Then we need to test this application in these operating systems.

Check for versions of operating system. Also we need to check what are devices for each operating systems in which it needs to compatible.

If client says it needs to be compatible with Mobile, Tablet & Laptop then we need to test application in these devices.

**30. How do you make sure you cover all the testing for given requirements.**

We do this by using Traceability Matrix where we map test cases against each requirement which are given by client.

**31**. What is difference between Sanity & Smoke Testing?

| **Smoke Testing** | **Sanity Testing** |
| --- | --- |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality / bugs have been fixed |
| The objective of this testing is to verify the "stability" of the system in order to proceed with more rigorous testing | The objective of the testing is to verify the "rationality" of the system in order to proceed with more rigorous testing |
| This testing is performed by the developers or testers | Sanity testing is usually performed by testers |
| Smoke testing is usually not documented or scripted | Sanity testing is usually documented |
| Smoke testing is a subset of Regression testing | Sanity testing is a subset of Acceptance 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 |