**Q.1.** **What is Software Testing?**

**Ans:-** [Software Testing](https://www.edureka.co/blog/software-testing-tutorial/#IntroductiontoSoftwareTesting)is a process of checking the completeness, correctness and quality of developed software with respect to the client’s expectations.

**Q2. Importance OF Testing:-**

Ans:- Used to verify whether the Software designed & developed as per

client requirement.

- It is used to find the defects in the software as early as possible.

- Used to improve the quality of Software.

- Used to improve the Customer Satisfaction.

- Used to gain the confidence in software before delivering to

customer.

- Helps for on time delivery of the product to the customer.

**Q3. What are the two main categories of software testing?**

**Ans:-** Software testing is a huge domain but it can be broadly categorized into two areas such as :

[**Manual Testing**](https://www.edureka.co/blog/what-is-manual-testing/)**:-** Manual software testing is a process where human testers manually run test cases, then generate the resulting test reports.

This is the oldest type of software testing where the testers manually execute test cases without using any test automation tools. It means the software application is tested manually by QA testers.

[**Automation Testing**](https://www.edureka.co/blog/automation-testing-tutorial/)**:-** It is a type of software testing where tester execute the test cases using automation tools.

The automation testing is used to change the manual test cases into a test script with the help of some automation tools.

Test Automation focuses on replacing manual human activity with systems or devices that enhance efficiency.

**Q4.Advantages and disadvantages of manual testing**

**Ans:-**

## Advantages of Manual Testing

* It does not require programming knowledge while using the Black box method.
* It is used to test dynamically changing GUI designs.

(If the Graphical User Interface of the Application changes dynamically then Test Automation is not recommendable.)

* Tester interacts with software as a real user so that they are able to discover usability and user interface issues.
* We can handle captcha, OTP, Google images manually.
* It ensures that the software is a hundred percent bug-free.
* It is cost-effective.
* Easy to learn for new testers.

## Disadvantages of Manual Testing

* It requires a large number of human resources.
* It is very time-consuming.
* Less accuracy (Human Users (Testers) may make mistakes, so we cannot expect more accuracy in Manual Testing.
* Tester develops test cases based on their skills and experience. There is no evidence that they have covered all functions or not.
* Test cases cannot be used again. Need to develop separate test cases for each new software.
* It does not provide testing on all aspects of testing.
* Since two teams work together, sometimes it is difficult to understand each other's motives, it can mislead the process.

**Q5. What different types of manual testing are there?**

Different types of manual testing are;

* + Black Box Testing
  + White Box Testing
  + Unit Testing
  + System Testing
  + Integration Testing
  + Acceptance Testing

## Q6.What are the levels of Software Testing?

In [software testing](https://www.javatpoint.com/software-testing-tutorial), we have four different levels of testing, which are as discussed below:

1. **Unit Testing**
2. **Integration Testing**
3. **System Testing**
4. **Acceptance Testing**

**Q7. What is Quality?**

**Ans:-**

* Product should meet customer requirement.
* On time delivery of product.
* Product designed and Developed as per Client Requirement.
* Sellable for reasonable cost.

**Software Quality:-**

Designed as per client expectations

**+** Meet Customer Expectations

**+** Within Cost **+** developed with in time + As per Testing Standard.

### Q8.  What is quality control and quality assurance?

**Ans:-** **Quality control** involves in product-oriented activities. It executes the program or code to identify the defects in the Software Application as well as making sure that the software meets all of the requirements put forth by the stakeholders.

**Quality Assurance** involves in process-oriented activities. It ensures the prevention of defects in the process used to make Software Applications. So the defects don’t arise when the Software Application is being developed.

### Q9. Diffrence Between QA and QC

### Ans:-

|  |  |
| --- | --- |
| Quality Assurance | Quality Control |
| QA is a set of activities for  ensuring quality in the processes by which products are developed.To prevent the defects when product is being developed. | QC is a set of activities for  ensuring quality in products. The  activities focus on identifying  defects in the actual products produce. |
| The goal of QA is to improve  development and test processes  so that defects do not arise when the product is being developed | The goal of QC is to identify  defects after a product is  developed and before it's released. |
| QA is a managerial tool. | QC is a corrective tool. |
| QA is process oriented. | QC is product oriented. |
| QA aims to prevent the defect. | QC aims to identify and fix defects |
| It is a method to manage the quality- Verification | It is a method to verify the quality- Validation |
| It's a Preventive technique | It's a Corrective technique |
| QA involves in full software development life cycle | QC involves in full software testing life cycle |
| It is performed before Quality Control | It is performed only after QA activity is done |
| It requires the involvement of the whole team | It requires the involvement of the Testing team |

### Q10.What is SDLC

Ans:-

SDLC is a process used by software industry to design,develop and test softwares.

The Software Development Lifecycle is a systematic process for building software that ensures the quality and correctness of the software built.

SDLC process aims to produce high-quality software which meets

customer expectations. The software development should be complete in the pre-defined time frame and cost.

\*/\*

**1. Requirement collection and analysis**:

The requirement is the first stage in the SDLC process. It is conducted

by the senior team members with inputs from all the stakeholders and

domain experts in the industry.

Planning for the quality assurance requirements and recognization of the risks involved is also done at this

stage.

This stage gives a clearer picture of the scope of the entire project and

the anticipated issues, opportunities, and directives which triggered the

project.

 All information gathered from this phase is critical to developing the product as per the customer requirements.

**Phase 2: Feasibility study:**

Once the requirement analysis phase is completed the next step is to define and document software needs. This process conducted with the help of 'Software Requirement Specification' document also known as 'SRS' document. It includes everything which should be designed and developed during the project life cycle.

There are mainly five types of feasibilities checks:

**• Economic:** Can we complete the project within the budget or not?

**• Legal:** Can we handle this project as cyber law and other

regulatory framework/compliances.

**• Operation feasibility:** Can we create operations which is

expected by the client?

**• Technical:** Need to check whether the current computer system

can support the software

**• Schedule:** Decide that the project can be completed within the

given schedule or not.

**Phase 3 : Design:**

In this third phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture. This design phase serves as input for the next phase of the model. There are two kinds of design documents developed in this phase:

**High-Level Design (HLD)**

• Brief description and name of each module

• An outline about the functionality of every module

• Interface relationship and dependencies between modules

• Database tables identified along with their key elements

• Complete architecture diagrams along with technology details

**Low-Level Design(LLD)**

• Functional logic of the modules

• Database tables, which include type and size

• Complete detail of the interface

• Addresses all types of dependency issues

• Listing of error messages

• Complete input and outputs for every module

**Phase 4: Coding:**

Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process. In this phase, Developer needs to follow certain predefined coding guidelines. They also need to use programming tools like compiler, interpreters, debugger to generate and implement the code.

**Phase 5: Testing:**

Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement. During this phase, QA and testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system.

**Phase 6: Installation/Deployment:**

Once the software testing phase is over and no bugs or errors left in the

system then the final deployment process starts. Based on the feedback

given by the project manager, the final software is released and checked

for deployment issues if any.

**Phase 7: Maintenance:**

Once the system is deployed, and customers start using the developed

system, following 3 activities occur

• Bug fixing - bugs are reported because of some scenarios which

are not tested at all

• Upgrade - Upgrading the application to the newer versions of the

Software

• Enhancement - Adding some new features into the existing

Software.

The main focus of this SDLC phase is to ensure that needs continue to be met and that the system continues to perform as per the specification

mentioned in the first phase.

**Q11. What Is Software Testing Life Cycle (STLC)**

Ans:- Software Testing Life Cycle refers to a testing process which has specific steps to be executed in a definite sequence to ensure that the quality goals have been met. In the STLC process, each activity is carried out in a planned and systematic way. All activities are done to improve the quality of the software product. Each phase has different goals and deliverables.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Comparison basis** | **SDLC** | **STLC** |
|  | **Representation** | SDLC stands for ***Software Development Life Cycle***. | STLC stands for ***Software Testing Life cycle***. |
|  |  | SDLC is mainly related to software development. | STLC is mainly related to software testing. |
|  | **Help in** | The SDLC will help us to develop a good quality software product. | The STLC will help in making the software bug-free. |
|  | **Performed** | The SDLC phases are done before the STLC phases. | The STLC phases are completed after SDLC phases. |
|  | **Focus on** | Besides the development phase, other phases like testing are also included. | The STLC concentrate only on testing the software. |
|  | **Resources** | While performing the SDLC process, we needed a greater number of developers to complete the development process. | The STLC process needed a smaller number of testers to complete the testing process. |
|  | **Goal** | Goal of SDLC is to complete successful development of software. | Goal of STLC is to complete successful testing of software. |
| **8.** | **Different phases** | The various phase includes in **Software Development Life Cycle** are as follows:   * Requirements Collection * Feasibility Study * Design * Programming or Coding * Testing * Installation * Maintenance | The various phase includes in **Software Testing Life Cycle** are as follows:   * Requirement collection or System study * Test Plan * Write test case * Traceability Matrix * Defect Tracking * Test Execution Report * Retrospective meeting |

**Waterfall Model** - refer in book

Que Explain Agile Methodology

Ans:-

**AGILE methodology** is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle.

1. Agile methodology is the combination of incremental and iterative approach to design software.
2. Agile methodology allows the team to work together more efficiently and effectively in developing complex projects.
3. Error can be fixed in the middle of the project.
4. Documentation attends less priority than software development.

**What is scrum?**

→ Its framework or procedure followed by the scrum team to develop application sprint wise.

**OR**

**SCRUM i**s an agile development method which concentrates specifically on how to manage tasks within a team-based development environment.

**What is sprint?**

→ It's the time duration taken by the scrum team to develop a few user stories.

Sprint duration will be 1 week to 4 week .

**Scrum Master**

Scrum Master is responsible for setting up the scrum team, conducting the sprint meeting and removes obstacles during the work progress.

**Product owner**

The Product Owner is responsible to creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration to the customer.

**Scrum Team**

Scrum Team manages its own work and organizes the work to complete the sprint or cycle.

**Product Backlog:**

Product Backlog is a repository where the list of Product Backlog Items stored and maintained by the Product Owner. The list of Product Backlog Items is prioritized by the Product Owner as high and low and also could re-prioritize the product backlog constantly.

**OR**

The Product Backlog is an ordered list of everything that is known to be needed in the product. It is the single source of requirements for any changes to be made to the product**. The Product Owner is responsible for the Product Backlog**, including its content, availability, and ordering.

**Sprint Backlog:**

Group of user stories which scrum development team agreed to do during the current sprint (Committed Product Backlog items)

**OR**

It is defined as collection of user stories with detailed explanation that scrum team is going to developed in current sprint.

**30 days plan ::**

It consist of requirement gathering and analysis, creating use cases before coding ,build code and during building daily scrum meeting and inspect or test the code over the sprint.

**Daily scrum meeting::**

It defined as meeting for task allocation , discussion among team member and scrum master regarding the work progress and obstacle during process.

**Sprint review Meeting::**

It defined as decision process to decide whether product completed over the sprint is potentially shippable or not by presenting product to product owner .

This is informal meeting .This is arrange by agile Scrum master.

When should we use Agile?

→ 1) For a long term project.

2) If the customer is not clear about requirements.

3) If a customer is expecting a product within a short period.

**Agile process:**

1. Customer or stack holder will prepare a requirement document or User stories and send it to the business analyst/ product owner.
2. BA will prepare SRS documents.
3. Stand up meeting will be conducted between stack holder and product owner.
4. In this meeting stack holder will prioritize user stories.
5. Product owners prepare product backlog as per stack holder priorities of user story.
6. Product backlog will be shared with the scrum team.
7. Scrum team will split product backlog into sprint backlogs.
8. Each sprint backlog will contain user stories that need to be completed in the sprint.
9. To develop a sprint scrum team will conduct a sprint analysis meeting.
10. In this meeting the scrum team will decides sprint duration by dividing each user story into tickets or units.
11. Designing use cases, coding and testing phase will execute parallely to develop the sprint.
12. Every day scrum master will conduct a **daily standup meeting** or

**scrum meeting.1814**

1. In this meeting every team member should explain what they did

yesterday, what the difficulties face and what they will do today.

1. At the end of the sprint scrum master will conduct a **sprint review**

**meeting.**

1. In this meeting the scrum team will give a live demo on sprint to

customers.

1. If the customer is satisfied, the product should be shifted.
2. Before developing the next sprint scrum master will conduct a sprint

**retrospect meeting**.

1. Every team member should explain any difficulties faced in the

previous sprint.

1. All the difficulties will be documented and identify solutions so that in the

next sprint development scrum team should not face difficulties.

**Advantages:-**

1) Backtracking is possible.

2) Cost of fixing bugs is low.

3) Good communication between customer and organisation.

4) Customers can easily track application progress.

5) Customers can use the product within a short period.

**Disadvantages:-**

1) Too much documentation.

2) Initial investment is high.

3) Application development will be delayed.

Que 88) What is the retrospect meeting?

→

**This meeting will be arranged by agile Scrum master.**

In this meeting the test lead will interact with every team member and

identifies all the drawbacks that are faced by the testing team.

All the difficulties will be documented in retrospect and store it in

QMS(Quality Management Services). This document will be helpful to test

leads for upcoming projects to prepare a test plan.

**V Model**

The V model is an extension of the waterfall model in which testing is done on each stage parallel with development in a sequential way. It is known as the **Verification and Validation Model.**

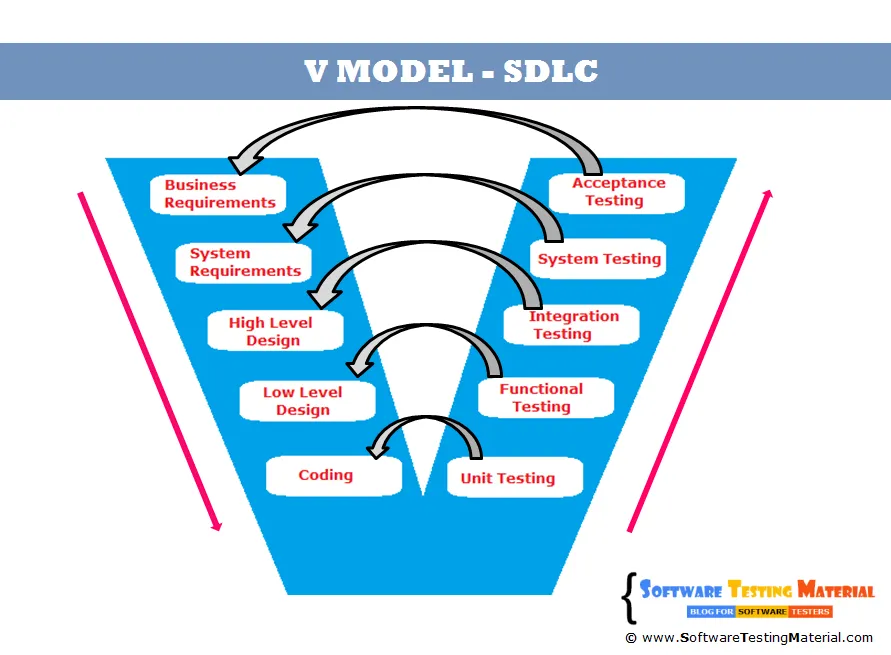
In waterfall model, testing starts only after development phase is completed.

The major drawback of waterfall model is we move to the next stage only when the previous one is completed and there is no change in the requirement in the middle of the project.

**But In the V model, the development and QA activities are done**

**Simultaneously.**

So V-Model contains Verification phases on one side of the Validation phases on the other side. Verification and Validation process is joined by coding phase in V-shape. Thus it is known as V-Model.



**1**. Once client sends BRS, both the teams (test and development) start their activities. The developers translate the BRS to SRS. The test team involves in reviewing the BRS to find the missing or wrong requirements and writes acceptance test plan and acceptance test cases.

**2**. In the next stage, the development team sends the SRS to the testing team for review and the developers start building the HLD (High Level Design Document) of the product. The test team involves in reviewing the SRS against the BRS and writes system test plan and test cases.

**3**. In the next stage, the development team starts building the LLD (Low Level Design) of the product. The test team involves in reviewing the HLD (High Level Design) and writes Integration test plan and integration test cases.

**4**. In the next stage, the development team starts with the coding of the product. The test team involves in reviewing the LLD and writes functional test plan and functional test cases.

**5**. In the next stage, the development team releases the build to the test team once the unit testing was done. The test team carries out functional testing, integration testing, system testing and acceptance testing on the release build step by step.

**Advantages:**

1. Testing starts in early stages of product development which helps to find the defects in the early stages
2. **Testing is involved in every stage of product development. It gives a quality product.**
3. Overall project cost is less.

Disadvantages:

1) Too much documentation.

2) Initial investment is high.

Que. When should we use the V and V model?

→ 1) For long term projects.

2) If the customer expects a high quality product.

## Verification testing:

It is also known as **static testing**, where we are ensuring that "**we are developing the right product or not**". And it also checks that in developed application all the requirements fulfilled or not given by the client.

## Verification testing includes different activities such as business requirements, system requirements, design review, and code walkthrough while developing a product.

## Validation testing:

Validation testing is also known as **dynamic testing**, where we are ensuring that **"we have developed the product is right."** And it also checks that the software meets the business needs of the client.

Validation testing is testing where tester performed functional and non-functional testing. Here **functional testing** includes [Unit Testing](https://www.javatpoint.com/unit-testing) (UT), [Integration Testing](https://www.javatpoint.com/integration-testing) (IT) and System Testing (ST), and **non-functional** testing includes User acceptance testing (UAT).

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Verification** | **Validation** |
|  | We check whether we are developing the right product or not. | We check whether the developed product is right. |
|  | Verification is also known as **static testing**. | Validation is also known as **dynamic testing**. |
|  | Verification includes different activities like Inspections, Reviews, and Walkthroughs. | Validation includes testing like [functional testing , system testing,](https://www.javatpoint.com/functional-testing)[integration and User acceptance testing.](https://www.javatpoint.com/integration-testing) |
|  | It is a process of checking the work-products (not the final product) of a development cycle to decide whether the product meets the specified requirements. | It is a process of checking the software during or at the end of the development cycle to decide whether the software follow the specified business requirements. |
|  | **Quality assurance** comes under verification testing. | **Quality control** comes under validation testing. |
|  | The execution of code does not happen in the verification testing. | In validation testing, the execution of code happens. |
|  | In verification testing, we can find the bugs early in the development phase of the product. | In the validation testing, we can find those bugs, which are not caught in the verification process. |
|  | Verification testing is executed by the Quality assurance team to make sure that the product is developed according to customers' requirements. | Validation testing is executed by the testing team to test the application. |
|  | Verification is done before the validation testing. | After verification testing, validation testing takes place. |
|  | In this type of testing, we can verify that the inputs follow the outputs or not. | In this type of testing, we can validate that the user accepts the product or not. |

**Static Testing:**

**Static testing is a verification process used to test the application without executing the code of the application. And it is a cost-effective process.**

[Static testing](https://www.javatpoint.com/static-testing) is a **verification** process

We can do some of the activities while performing static testing such as code Review, Walk Through, Inspection, and Analysis.

#### Note: Static testing is performed in the white box testing phase, where the developer checks every line of the code before giving it to the Test Engineer.

**Dynamic Testing:**

In dynamic testing, testers execute the test cases by giving input values and verify the expected result with the actual result.

**Dynamic testing** is a **validation** process.

It is a **validation process** where functional testing [unit, integration, system, and user acceptance testing] and non-functional testing [Performance, usability, compatibility, recovery and security testing] are performed.

**Exploratory testing :**

1. This process will be carried out by domain experts.
2. They perform testing just by exploring the functionalities of the application without having the knowledge of the requirements.
3. **Exploratory testing will perform when the requirement is missing or requirement doesnot exist.**

## How to perform exploratory testing

To perform exploratory testing, first, we will start using the application and understand the requirement of the application from the person who has good product knowledge such as senior test engineer, and developers.

Then we will explore the application and write the necessary document, and this document is sent to the domain expert, and they will go through the document.

And we can test the application based on our knowledge, and taking the help of the competitive product, which is already launched in the market.

#### ****GUI Testing****

Graphical User Interface Testing is to test the interface between the application and the end user.

**Levels of Testing:**

# Unit , integration ,system, User acceptance Testing

* **Unit Testing** used to test single software or to test individual functionality of the application.
* **Integration Testing** used to test a group of units of software,
* **System Testing** used to test a whole system and
* **Acceptance Testing** used to test the acceptability of business requirements.

# 1.Unit Testing

1. Unit testing involves the testing of each unit or an individual component of the software application. It is the first level of functional testing. The aim behind unit testing is to validate unit components with its performance.
2. A unit is a single testable part of a software system and tested during the development phase of the application software.
3. White box testing approach used for unit testing and usually done by the developers.
4. **Whenever the application is ready and given to the Test engineer, he will start checking every component of the module or module of the application independently or one by one, and this process is known as Unit testing or components testing.**

* Unit testing helps tester and developers to understand the base of code that makes them able to change defect causing code quickly.
* Unit testing helps in the documentation.
* **Unit testing fixes defects very early in the development phase that's why there is a possibility to occur a smaller number of defects in upcoming testing levels.**
* It helps with code reusability by migrating code and test case.

**Integration testing**

1. In integration testing, testing data flow between two features based on relation.
2. Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as **integration testing**.
3. **In this level of testing, testing team will verify how the modules are interact with each other and sharing the common data.**

For ex- in case of flipkart online product purchase at that time user register by using username, password, mobile no., emailid etc and save into database and at the time of login user have to used username, password details which he already enter at the time of registration.

In this scenario, login and registration screen sharing the common data.

**System Testing**

1. To check the end-to-end flow of an application as a user is known as **System testing.**
2. System Testing includes testing of a fully integrated software system.
3. In this level of testing, testing team will check system behavior, functionality, features of the application is called **system testing.**

For example, verify that user can able to purchase product or not.

1. It **is end-to-end testing** where the testing environment is similar to the production environment.
2. **System testing falls under Black box testing**

**User Acceptance Testing**

1. In this level of testing, end user or customer verify the application behavior along with testing team.
2. User are checking end to end functionality of the application.

It is divided into two types

1. Alpha testing
2. Beta testing
3. **Alpha testing**: When end user and customer verify the application functionality at developer location is called alpha testing.
4. **Beta testing:** Once the alpha testing done then will perform beta testing.

It will perform in client location and verify all the functionality of application according to SRS or BRD document.

**Type of testing:**

**1. Functional**

**2. Non-functional**

**Functional:**

1. It is a type of software testing which is used to verify the functionality or behavior of the software application.
2. Functional testing is nothing but how your feature should work. It mainly focuses on feature working or not. It is given by customer.
3. In functional testing, each function tested by giving the value, determining the output, and verifying the actual result with the expected result.

**Types:**

1. Smoke Testing
2. Sanity Testing
3. Re- Testing
4. Regression Testing
5. Adhoc testing
6. Monkey testing
7. Buddy testing
8. Maintenance Testing
9. User Interface Testing

**Non-functional:** it mainly focuses on performance, load, security, reliability and scalability of software. We do non-functional testing to give high quality software.

For non-functional testing we need manual +automation testing.

Types

1. Performance Testing
2. Load Testing
3. Stress Testing
4. Volume Testing
5. Failover Testing
6. Reliability Testing
7. Documentation Testing
8. Localization Testing
9. Installation Testing

## What is Build?

**In manual testing, build is software that contains some set of features/bugs and it is delivered to the testing team that needs to be tested for the product's stability.**

In other words, we can say that the build is used to change the code into the application format. Every new build will be the improved version of the new build.

## What is Release?

The **Release** is a final product or a project, which is delivered to the customer.

It involves the complete activities from the **Requirement, Designing, Development, and Testing phases** until it is handed over to the customer.

In other words, we can say that a release is an entirely developed application, while the build is the part of an application or the software.

#### Note:  Whenever a build is tested and specialized by the testing team, it is handed over to the clients as Release.

**Smoke Testing:**

1. Testing the basic & critical feature of an application is known as **smoke testing.**
2. To perform Smoke testing we use only valid data. So it is also known

As Positive Testing.

1. Smoke Testing is done to make sure if the build we received from the development team is testable or not, hence it is also known as **Build Verification Testing.**
2. When we perform smoke testing, we can identify the blocker bug at the early stage.
3. It is a subset of acceptance testing.

## When we perform smoke testing

Generally, whenever the new build is installed, we will perform one round of smoke testing because in the latest build, we may encounter the blocker bug.

**Why should we perform Smoke testing?**

1) To identify blocker bugs at initial state.

2) To insure the build is testable or not.

3) To avoid increasing project duration and cost.

4) To reduce tester efforts.

5) To avoid delay in new build testing.

**Sanity Testing:**

1. Sanity Testing is done during the release phase to check for the main functionalities of the application without going deeper.
2. Sanity testing ensures that the newly added features may not affect or impact on the functionalities of existing features.
3. It is a subset of **regression testing**, which mainly focus on the important features of the application.
4. Sanity testing cannot be documented.
5. Sanity testing is done by the test engineers.

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Smoke** | **Sanity** |
| **1.** | In smoke testing, Testing the basic & critical feature of an application is known as **smoke testing.** | Sanity Testing is done during the release phase to check for the main functionalities of the application without going deeper. |
| **2.** | In smoke we perform only positive testing. | In sanity we perform positive and negative testing. |
| **3.** | Smoke testing is scripted. | Sanity testing is unscripted. |
| **4.** | It’s wide and shadow testing | It's narrow and deep testing. |
| **5.** | Smoke testing is documented. | Sanity testing is not documented. |
| **6.** | It is performed by both testers and developers. | It is performed by only testers. |
| **7.** | It is done at the “build level”. | It is done at the “release level” |
| **8.** | It is used to test End to End function of the application. | It is used to test only modified or defect fixed functions. |
| **9.** | It is considered as a subset of **acceptance testing.** | It is considered as a subset of **regression testing.** |

## Re-testing:

1. To ensure that the defects which were found in the earlier build were fixed or not in the current build. Say, Build 1.0 was released. Test team found some defects (Defect Id 1.0.1, 1.0.2) and posted. Build 1.1 was released, now testing the defects 1.0.1 and 1.0.2 in this build is retesting.

**2. Verifying or checking bugs is really fixed or not.**

**Process:**As per the Bug Life Cycle, once a tester found a bug, the bug is reported to the Development Team. The status of Bug should be “New”. The Development Team may accept or reject the bug. If the development team accepts the bug then they do fix it and release it in the next release. The status of the bug will be “Ready For QA”. Now the tester verifies the bug to find out whether it is resolved or not. This testing is known as **retesting**. Retesting is a planned testing. We do use same test cases with same test data which we used in the earlier build. If the bug is not found then we do change the status of the bug as “Fixed” else we do change the status as “Not Fixed” and send a Defect Retesting Document to the development team.

**Regression Testing:**

1. Regression testing is a type of software testing to ensure that the newly added features may not affect the functionalities of current or existing features.
2. Regression Testing is done to verify that a code change in the software does not impact on the existing feature of the application.
3. In Regression Testing test cases are re-executed in order to check whether the previous functionality of the application is working fine and the new changes have not introduced any new bugs.

|  |  |  |
| --- | --- | --- |
| Sr.No. | **Re-testing** | **Regression Testing** |
| 1. | Re-testing is to ensure that the defects which were found and posted in the earlier build were fixed or not in the current build. | Regression testing is a type of software testing to ensure that the newly added features may not affect the functionalities of current or existing features. |
| 2. | Re-testing is done on the basis of the[Defect](https://www.guru99.com/defect-management-process.html)fixes | The purpose of Regression Testing is that new code changes should not have any side effects to existing functionalities |
| 3. | **Retesting is done only for failed test cases** | **Regression testing is done for passed test cases** |
| 4. | **Re-testing makes sure that the bug is fixed or not** | **Regression testing checks for unexpected side-effects** |
| 5. | Re-testing executes a defect with the same data and the same environment with different inputs with a new build | Regression testing is only done when there is any modification or changes become mandatory in an existing project |

**Adhoc Testing**

1. Testing application randomly without referring to any requirement document.
2. Testing an application without using any logic.
3. If we identify more creative scenarios but time is not sufficient perform adhoc testing at the end of next test cycle.
4. In adhoc testing not maintain the documentation, no creating any test cases, tester check the functionality of the application manually

**Monkey Testing**

**Monkey Testing** is a software testing technique in which the tester enters any random inputs into the software application without predefined test cases and checks the behavior of the software application, whether it crashes or not.

**Buddy Testing**

If any developer design and develop any application features and asked one of the tester to verify that feature against the requirement this type of testing is called buddy testing.

**Test cases design techniques**

Two types of test cases design technique:

White box testing

Black box testing

**White box testing:-**

* It is also called as glass box **testing, clear box testing, open box testing and structural testing.**
* White Box Testing is based on the application’s internal code structure. In this type of testing programming skills are required to design test cases.
* **It tests internal coding and infrastructure of a software** focus on checking of predefined inputs against expected and desired outputs.
* The primary goal of white box testing is to focus on the flow of inputs and outputs through the software and strengthening the security of the software.

Developers do white box testing. In this, the developer will test every line of the code of the program. The developers perform the White-box testing and then send the application or the software to the testing team, where they will perform the [black box testing](https://www.javatpoint.com/black-box-testing) and verify the application along with the requirements and identify the bugs and sends it to the developer.

The developer fixes the bugs and does one round of white box testing and sends it to the testing team. Here, fixing the bugs implies that the bug is deleted, and the particular feature is working fine on the application.

**White Box Testing Techniques:**

1. Statement Coverage
2. Branch Coverage
3. Condition Coverage
4. Multiple Condition Coverage
5. Path Coverage
6. Control flow testing
7. Data flow testing

#### ****Statement coverage:****

It makes sure that each line of source code has been executed and tested.

#### ****Decision coverage:****

It ensures that every decision (true/false) in the source code has been executed and tested.

#### ****Path coverage:****

It ensures that every possible route through a given part of the code is executed and tested.

**Black box testing**

* Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding.
* In this type of testing team verify the system behaviour by providing input to the system without having access of code based on specification.

1. Error guessing technique
2. Equivalence partitioning
3. Boundary value analysis
4. State transition technique
5. Decision table based testing
   1. **Error guessing technique**

:- This technique is based on the experience where the test analysts use their experience to guess the problematic part of the testing application. Hence, the test analysts must be skilled and experienced for better error guessing.

The technique counts a list of possible errors or error-prone situations. Then tester writes a test case to expose those errors. To design test cases based on this software testing technique, the analyst can use the past experiences to identify the conditions.

* 1. **Equivalence partitioning**

:- It is also known as equivalence classing in this technique we divided data into classes (such as valid class and invalid class).

We are dividing whole range in different classes and one value is representing to whole class.

**Example:**

Suppose I have one text field in an application and my requirement says that it can accept numeric digit from 1 to 500 So instead of putting value from 1 to 500 in text field we entered only 6 value from each class if that value will get accepted then it means whole class will get accepted.

Different classes as below

• -100 to 0 -5 not accept whole class is accepted

• 0 to 100 6 accept

• 101 to 200 109 accept

• 201 to 300 299 accept

• 301 to 400 311 accept

• 401 to 500 479 accept

• 501 to 600 511 not accept whole class is not accepted

**Boundary value analysis**

In this technique we check data only at boundaries and just below and above the boundary.

This technique is used for the text box accepting data in the form of range so instead of writing whole range data we only write six test cases. It includes maximum, minimum, inside or outside boundaries, typical values and error values.

It is generally seen that a large number of errors occur at the boundaries of the defined input values rather than the center.

**Example:**

Suppose I have one text field in an application and my requirement

says that it can accept numeric digit from 1 to 10

Then we can say our lower boundary is A and upper boundary is B

then

Boundary+1 is maximum and boundary -1 is minimum value

|  |  |  |  |
| --- | --- | --- | --- |
| **Boundary A** | | **Boundary B** | |
| **A-1** | 0 | B-1 | 9 |
| A | 1 | B | 10 |
| A+1 | 2 | B+1 | 11 |

**State Transition technique**

When the application under case changes it’s behaviour or the states after receiving the user input then state transaction technique will be used.

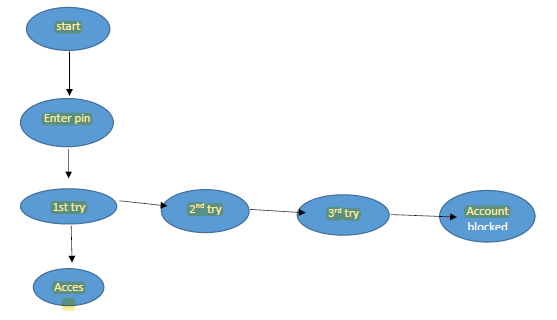
The tester can perform this action by entering various input conditions in a sequence. In State transition technique, the testing team provides positive as well as negative input test values for evaluating the system behaviour.

**Guideline for State Transition:**

1. State transition should be used when a testing team is testing the application for a limited set of input values.
2. The technique should be used when the testing team wants to test sequence of events which happen in the application under test.
3. E.g---we open new company food delivery
4. Requirement--- after 3rd atttempt of login information we will block user if it is wrong information.

|  |  |  |  |
| --- | --- | --- | --- |
| state | Login password | Correct Incorrect | Incorrect password |
| S1 | 1st attempt | S4 | S2 |
| S2 | 2nd attempt | S4 | S3 |
| S3 | 3rd attempt | S4 | S5 |
| S4 | Order food |  |  |
| S5 | Account block msg |  |  |

In the above-given table when the user enters the correct PIN, the state is transitioned to Access granted. And if the user enters an incorrect password, he or she is moved to next state. If he does the same 3rd time, he will reach the account blocked state.



**Decision table based testing**

A decision table is also known as to Cause-Effect table. This software testing technique is used for functions which respond to a combination of inputs or events.

For example, a submit button should be enabled if the user has entered all required fields. The first task is to identify functionalities where the output depends on a

combination of inputs. If there are large input set of combinations, then divide it into smaller subsets which are helpful for managing a decision table.

For every function, you need to create a table and list down all types of combinations of inputs and its respective outputs. This helps to identify a condition that is overlooked by the tester.

**Following are steps to create a decision table:**

• Enlist the inputs in rows

• Enter all the rules in the column

• Fill the table with the different combination of inputs

• In the last row, note down the output against the input combination.

**Example:**

we open one food delivery company and my conditions are

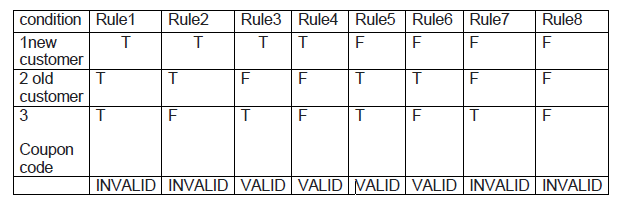
INPUT---CONDITIONS

New customer---15%

Old customer---10%

Coupon code---20% ----conditions

Test cases = number of rules= 2^no.conditions = 2^3 =8



|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **White box Testing** | **Black box Testing** |
|  | The developers can perform white box testing. | The test engineers perform the black box testing. |
|  | To perform WBT, we should have an understanding of the programming languages. | To perform BBT, there is no need to have an understanding of the programming languages. |
|  | In this, we will look into the source code and test the logic of the code. | In this, we will verify the functionality of the application based on the requirement specification. |
|  | In this, the developer should know about the internal design of the code. | In this, there is no need to know about the internal design of the code. |
|  | It is also called as clear box testing. | It is also called closed testing. |
|  | Types of White Box Testing:  • Path Testing  • Loop Testing  • Condition testing | Types of Black Box Testing:  • Functional Testing  • Non-functional testing  • Regression Testing |

Que 35) Explain Defect Life Cycle

1. When the tester identifies a bug prepare a bug report with status **new** and send it to the development team.
2. Developer lead will review bug reports, identify developers who develop that feature and assign bug reports by changing status **Assign**.
3. Developers will **review**/**open** bug reports, reproduce a bug and modify source code, change bug report status to **Fixed**.
4. Then new build send to the testing team.
5. When a new build comes for testing the tester will recheck bug fixes or not if it is fixed change status to **Closed**, if bug is not fixed to Reopen and send it to develop lead.
6. This process continues until the bug is fixed.

#### ****Defect****

The variation between the actual results and expected results is known as a **defect**. If a developer finds an issue and corrects it by himself in the development phase then it’s called a **defect**.

#### ****Bug****

If testers find any mismatch in the application/system in testing phase then they call it as **Bug**.

If a developer accepts a tester identified defect it will be

considered as Bug.

#### ****Error****

Error is nothing but the human mistakes. Error is committed by the developer.

We can’t compile or run a program due to a coding mistake in a program. If a developer unable to successfully compile or run a program then they call it as an error.

#### ****Failure****

Once the product is deployed and customers find any issues then they call the product as a failure product. After release, if an end user finds an issue then that particular issue is called as a failure.

#### ****Bug Leakage****

**Bug identified by end user or customer while using application.**

A bug which is actually missed by the testing team while testing and the build was released to the Production. If now that bug (which was missed by the testing team) was found by the end user or customer then we call it as Bug Leakage.

Bug leakage = No. of defects found in UAT

No. of defects found in QA testing

#### ****Bug Release****

Releasing the software to the Production with the known bugs then we call it as Bug Release. These known bugs should be included in the release note.

Bug identified by tester not fixed by developer and known by

customer.

**Test Strategy:**

Test Strategy is a high level document (static document) and usually developed by project manager. It is a document which captures the approach on how we go about testing the product and achieve the goals. It is normally derived from the Business Requirement Specification (BRS). Documents like Test Plan are prepared by keeping this document as base.

**Test Plan:**

In test planning, QA manager will start to write test plan and test plan determines future related activities.

In test plan QA manage calculates the estimated effort and cost for the testing work.

• Identify types and which test we need to perform

• Gather all data which help in testing

• Test planning –future testing related activity--- how many days are

required to perform test

Activities are carried out in test plan are

:-Test tool selection, test effort,resourse planning training

requirement

**Exit criteria** : test plan document and effort estimation.

**Test Scenario**

A **Test Scenario** is defined as any functionality that can be tested. It is also

called *Test Condition* or *Test Possibility*.

Test Scenario gives the idea of what we have to test. Test Scenario is like a high-level test case.

**Test Case**

A **Test Case** is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, post condition developed for specific test scenario to verify any requirement.

Test Case is mostly derived from test scenarios while Test Scenarios

are derived from test document like BRS and SRS.

**OR**

**Test cases** are the set of positive and negative executable steps to verify the functionality of the applications. Test case has a set of test data, test steps, pre-conditions, post-conditions, expected result, actual results.

**Test data:** Test data is the data that is used by the testers to run the test cases. Whilst running the test cases, testers need to enter some input data. To do so, testers prepare test data. It can be prepared manually and also by using tools.

For example, To test a basic login functionality having a user id, password fields. We need to enter some data in the user id and password fields. So we need to collect some test data.

**Test Envornment:** Test Environment is the combination of hardware and software on which Test Team performs testing.

Example:

* Application Type: Web Application
* OS: Windows
* Web Server: IIS
* Web Page Design: Dot Net
* Client Side Validation: JavaScript
* Server Side Scripting: ASP Dot Net
* Database: MS SQL Server
* Browser: IE/FireFox/Chrome

**Test bed**

:- It’s combination of test data and test environment.

An environment configured for testing. Test bed consists of hardware, software, network configuration, an application under test, other related software.

#### ****Test coverage****

Test coverage helps in measuring the amount of testing performed by a set of tests.  
Test coverage can be done on both functional and non-functional activities. It assists testers to create tests that cover areas which are missing.

#### ****Test Closure****

Test Closure is the note prepared before test team formally completes the testing process. This note contains the total no. of test cases, total no. of test cases executed, total no. of defects found, total no. of defects fixed, total no. of bugs not fixed, total no of bugs rejected etc.,

#### ****What is RTM?****

Requirements Traceability Matrix (RTM) is used to trace the requirements to the tests that are needed to verify whether the requirements are fulfilled. We have to ensure that every requirement has atleast 1 test case. Requirement Traceability Matrix AKA Traceability Matrix or Cross Reference Matrix.

#### ****What is Test Metrics?****

Software test metrics is to monitor and control process and product. It helps to drive the project towards our planned goals without deviation. Metrics answer different questions. It’s important to decide what questions you want answers to.

#### ****List out Test Deliverables?****

1. Test Strategy
2. Test Plan
3. Effort Estimation Report
4. Test Scenarios
5. Test Cases/Scripts
6. Test Data
7. Requirement Traceability Matrix (RTM)
8. Defect Report/Bug Report
9. Test Execution Report
10. Graphs and Metrics
11. Test summary report
12. Test incident report
13. Test closure report
14. Release Note
15. Installation/configuration guide
16. User guide
17. Test status report
18. Weekly status report (Project manager to client)

**Defect Severity:**

**Defect severity can be defined as the impact of the bug on customer’s business.** Severity is a parameter to denote the implication and the impact of the defect on the functionality of the software.

**Severity** is set by Tester.

In simple words, how much effect will be there on the system because of a particular defect. It can be **critical, high, medium and low**.

1. **Critical:** if the severity of a bug is a Critical, which means we cannot proceed to the next module, and unnecessarily test engineer sits ideal.

**For example-**

**A major feature is not working:** Login to HDFC, amount transfer is not working

**The major flow is not working:** Login and signup itself not working in HDFC application.

1. **High:** if it is High, that means the main functionality is not working, and the test engineer cannot continue testing.

**Ex-**in flipkart website user is unable to search product.

**-**Wrong logo displayed on webpage so it is impact on the customer business.

1. **Medium:** if it is Medium, which means that the supporting components and modules are not working fine, but test engineer can continue the testing.

**For example-**sms notification isnot received by the customer after the booking of the ticket on msrtc website but mail notification is working.

1. **Low:** if the severity of a bug is low, which means that all the U.I problems are not working fine, but testing can be processed without interruption.

**For example-** defect related to spelling mistake, grammatical mistakes in the webpage.

Note- According to defect severity will decide defect priority and according to defect priority developer will work.

**Defect priority:-**

**Priority:** This section explains how fast tester identified bug need to be fixed by developer.

Defect priority can be defined as how soon the defect should be fixed. **It can be very High, High, Medium or Low**.

**Defect having the higher priority should be fixed first.**

# **Note** : **Severity** is set by Tester and **Priority** is set by Developers

Sometimes tester can also set.

**Priority types**

a)Very High : If bugs need to be fixed in the same build.

b) High : Bug needs to be fixed in the next build.

c) Medium : Bug needs to be fixed within 5 to 10 builds.

d) Low : Bug needs to be fixed in any build even delivering product to

Customer.

**Examples-**

**High Priority & High Severity:**Submit button is not working on a login page and customers are unable to login to the application

**Low Priority & High Severity:**Crash in some functionality which is going to deliver after couple of releases

**High Priority & Low Severity:**Spelling mistake of a company name on the homepage

**Low Priority & Low Severity:**FAQ page takes a long time to load

Que. Explain High severity with a high priority with example.

:- The identified bug will show a savior impact on customer business and the number of users using the application in bug identified ways are more.

\***Bug**: User unable to login app.

Severity: Critical/High

Priority: High

Que.Explain High severity with a low priority with example.

:-The identified bug will show saviour impact on customer business and number of users uses applications in bug identified ways are very rare/less.

\***Bug**: user unable to login using help option.

Severity: High

Priority: Low

Que. Explain Low severity with a high priority with example.

:- The identified bug will not show any impact on customer business but every user can easily understand that mistake.

\***Bug**: Logo contains mistakes.

Severity: Low

Priority: High

Que. Explain Low severity with a Low priority with example.

:- Identified bugs will not show any impact on customer business no user can understand mistakes are less.

\***Bug**: Spelling mistake grammatical mistakes.

Severity: Low

Priority: Low

|  |  |  |
| --- | --- | --- |
| Sr.no. | **Severity** | **Priority** |
|  | Defect severity can be defined as the impact of bug on customer’s business.  Ex-wrong logo displayed on webpage | Defect Priority can be defined how fast tester identified bug need to be fixed by developer. |
|  | Severity is set by Tester(Team lead) | Priority is set by Developers |
|  | It can be **critical, high, medium and low**. | It can be urgent, High, Medium or Low. |
|  | According to severity, priority will decide. | According to priority, developer will work. |