**What is software Testing?**

Testing is a process of validating whether the developed software is working as per the customer requirements or not and with an intention of finding defects.

**What to Test?**

Functionality – Verifying whether all functions are working or not.

Appearance (Look) (UI Testing) – By seeing the beauty of the application, we can decide its good or not.

User Friendliness (Usability Testing) – Easy to understand and easily operated or not.

Security – Whether all security conditions are observed correctly or not.

Load Testing – Multiple users are capable of handling one application at a time.

Multiple Device Support – A single application is operated in different devices or not.

Multiple OS/Browser Support.

Speed (Performance testing) – Quick response.

**What are the principles of Software Testing?**

There are seven principles of testing. They are

* Shows presence of defects
* Early testing
* Defect Clustering
* Exhaustive testing is impossible.
* Pesticide Clustering
* Testing is context depending.
* Absence of error fallacy

**Types of testing?**

**Manual Testing:** Any Testing activity carried out with man effort without any coding technology and tools.

**Automation Testing**: Testing s/w using some automation tools and testers execute the test scripts and obtain the results. 🡪 Some of the testing tools are QTP/UFT and selenium.

**What types of applications we test?**

Mobile Applications, Web applications.

**Testing Methods?**

🡪Static Testing Method (Verification) – This test is conducted with the help of Reviews and Walkthroughs.

🡪Dynamic Testing Method (Validation) – White box, black box, Grey box testing.

**What is a Review?**

Examining any project related work is called a review. Four types: Requirement review (By BA), Design review (SA), Code review (Developers), Test case review (Testers).

* If a review is done by following a plan, process and documents is called “formal reviews”. “Inspection” (done while doing a work) and “audit” (After the completion of the work) are called formal reviews.
* If a review is done without any process is called “Informal review”. “Peer Review” is conducted among the team members informally.

**What is an error?**

Any human incorrect actions that produce wrong output is called error.

**What is Defect/ Fault/Bug?**

Deviation between the expected behavior to the actual behavior of the system is called fault.

**What is failure?**

Failure is a defect that occurs in software because of the error made by human.

**What is SDLC?**

Software Development Life Cycle(SDLC) is a process of developing a system in a planned cycle as per the customer expectations and is less expensive to maintain. In these each phase delivers requirements to the next phase as per the cycle.

**Phases of SDLC:**

Requirements Gathering - BA is responsible – BRS (Business Requirement Specifications0 documents are prepared by the BA. – SRS (Software Requirement Specifications).

Planning/Analysis – Manager

Designing – HLD (High level Design specifications) are available

LLD (Low level Design Specifications).

Coding – UI/Applications developers(java)/ Data base developers involved.

Testing

Deployment/Delivery and maintenance.

**Types of SDLC models:**

🡪Water fall model

🡪Spiral model

🡪V model

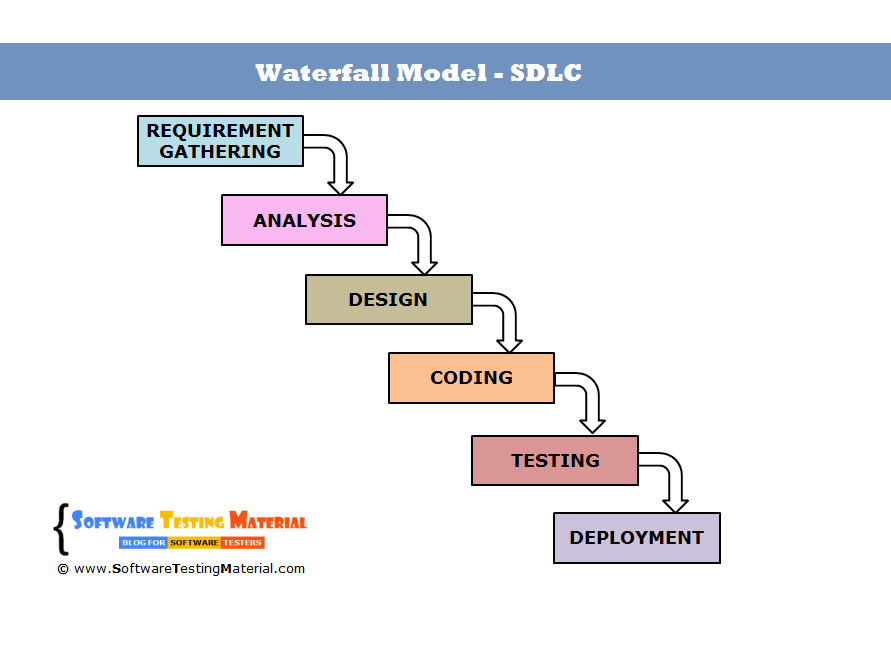
🡪Prototype model

🡪Agile model

**Waterfall model in SDLC?** 🡪Classic Approach.

This waterfall model is a flowing downwards method where next phase begun once the goal of the previous phase is completed.

🡪This method is mainly applicable for short term projects (i.e. Calculator, Attendance Management).



Advantages: This model keep records of the complete process that may be useful for future cause. It is very simple to design. Complete planning and finalizing the requirements are done at the beginning itself therefore a stable product is obtained.

Drawbacks: Once one phase is completed we cannot go back to it. Any changes made in requirements may lead to change in coding and designing and failure of project. Not applicable for long projects.

**Types of software Testing?**

**Functional Testing**: Main aim of these is to test the functionality of any new developed s/w application. For example: when we enter correct input, are we getting the actual output or not, and this is like black box testing and here there is no matter of what the coding is.

**Non-functional Testing**: Here testing is done for the system to identify how well the performance, load, stress, security and compatibility of the application is happening. And testing the quick response of any request.

**Integration Testing**:  Integration Testing is the process of testing the interface between the two software units. Integration testing is done by multiple approaches such Big Bang Approach, Top Down Approach (done by dummy modules called studs and Drivers), Bottom-Up Approach and Hybrid Integration approach.

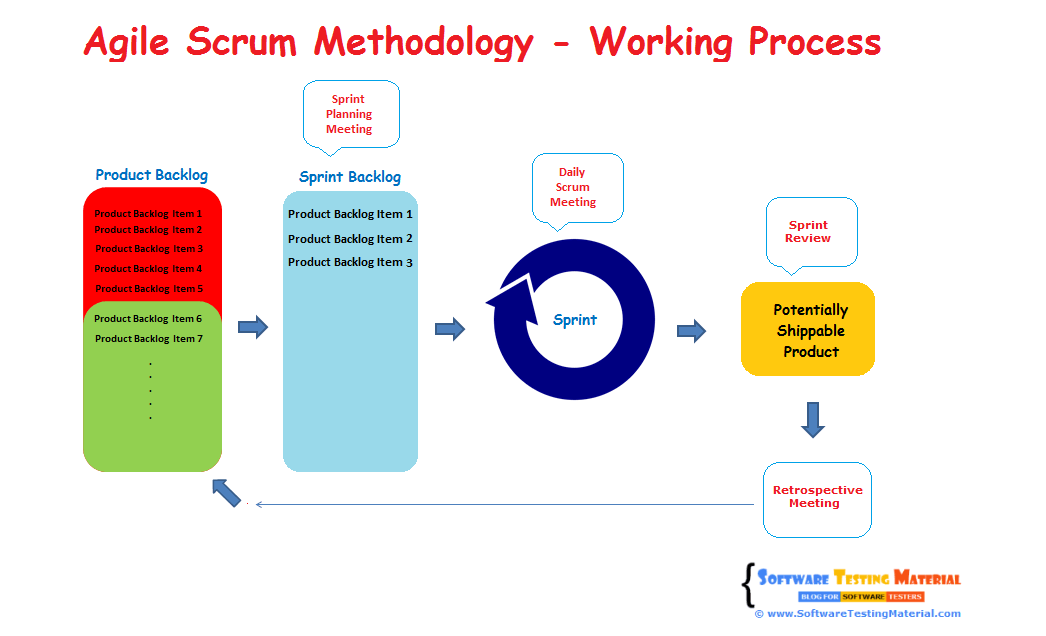
**What is a Stub and Driver?**

Stubs are “called programs” usually applicable in Top-down Approach.

Whereas Driver are “Calling programs” usually applicable in Bottom-up approach.

For example, whenever we need to test “Login Page” and “Admin Page”, Login page calls up admin page for testing once it is done, but if the admin page is not yet ready for testing, then we create a dummy module called “Studs” instead of Admin page. If login page is not ready for testing, then we use “Driver”.

**Agile Scrum Methodology?**



**What is Product Backlog item (PBI)?**

Product backlog item ("PBI", "backlog item", or "item") is a unit of work small enough to be completed by a team in one Sprint iteration. Backlog items are decomposed into one or more tasks.

**What is user story/feature/sprint backlog item and tasks in user story?**

The **sprint backlog** is a list of tasks identified by the Scrum team to be completed during the Scrum **sprint**. During the **sprint** planning meeting, the team task is to select some number of **product backlog items**, usually in the form of **user stories**, and identifies the tasks necessary to complete each **user story.**

**What is sprint planning meeting?**

Sprint planning meeting is a timeboxed working session where all the team members agreed to complete set of backlog items based on the velocity and capacity of the team and length of the sprint.

**What is sprint review meeting (SRM)?**

Sprint review meeting is held by the scrum meeting where they show their accomplished tasks during sprint like a demo for the new features.

**What is sprint Retrospective?**

During the Sprint Retrospective, the team members about how well went in the Sprint and discusses about what could be improved in the next Sprint.

**What is TDD and BDD?**

**TDD:** TDD or Test-Driven Development is a process where you write and run your tests.It also reduces the chances of having bugs in your tests.

The TDD process consists of the following steps:

* Start by writing a test
* Run the test and any other tests. At this point, your newly added test should fail. If it doesn’t fail here, it might not be testing the right thing and thus has a bug in it.
* Write the minimum amount of code required to make the test pass.
* Run the tests to check the new test passes.
* Optionally refactor your code.
* Repeat from 1.

**BDD**: Behavior Driven Development or BDD, when applied to automated testing, it is one of best practices for writing great tests. BDD suggest testing the behaviors, so instead of thinking of how the code is implemented, we spend a moment thinking of what the scenario is about.

**Levels of Testing?**

Unit Testing

Integration Testing

System Testing

Acceptance Testing.

**Unit Testing** (AKA or Component Testing): Done by the developers in their environment to check each individual module of the source code is working properly or not.

# System Testing: It’s a black box testing. Testing the fully integrated application this is also called as end to end scenario testing. To ensure that the software works in all intended target systems.

# Acceptance Testing: Pre-production testing. It is the sign off testing done by the end user along with tester to validate the functionalities of the application. Types of Acceptance testing are alpha (usability testing done by clients/ outsiders with developers), beta (end user testing), Gamma Testing (direct testing without any in-house testing activities).

# Manual Testing Vs Automation Testing?

# Manual Testing: Done manually to find defects without any automation tools knowledge. Test cases are executed to find results.

# Pros: It is applicable for all applications and useful for small life cycle applications. Less time and cheaper in terms of initial investments, can do adhoc testing. Cons: More expensive and time consuming.

# Automation Testing: Testing s/w application using automation tools to find defects. Test scripts are executed to find results.

# Pros: faster and cheaper compared to manual testing. More reliable, and powerful. Cons: Only for stable products, needed to maintain a lot.

# Black Box Testing?

# (Behavioral or Input-output Testing): Black Box Testing is a [software testing](https://www.softwaretestingmaterial.com/software-testing/) method in which testers evaluate the functionality of the software under test without looking at the internal code structure.

# White Box Testing (Glass or clear box testing): White Box Testing is based on applications internal code structure. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.

# Smoke and Sanity Testing?

# Smoke Testing: Smoke test is to make sure whether the build given by the development team is testable or not and is also called “Day 0” or “Build Level” Testing.

# Sanity Testing: This testing is done at the release of the application and checks only the main functionalities without going deeper into it and is also called “Release Level Testing”.

# What is Regression Testing & When do we Do?

# Regression testing is a repeated testing process for already tested code after modification to find defects due to s/w changes. Usually we do these when we modify any application.

# What is Retesting & When do we Do?

# Retesting is running the previously failed test cases again on the new software to verify whether the defects posted earlier are fixed or not. Usually we do these when bug fix is specified in release note or when a client calls for retesting.

# Difference between Regression Testing and Retesting?

# Retesting done on failed test cases whereas Regression Testing done on passed test cases.

# Retesting makes sure that the original defect has been corrected whereas Regression Testing makes sure that there are no unexpected side effects.

# Entry and Exit Criteria?

# Entry Criteria: The prerequisites that must be achieved before commencing the testing process.

# Exit Criteria: The conditions that must be met before testing should be concluded.

# Test Scenario Vs Test Cases?

# Test Scenario gives the idea what to be tested by using Test Scenario Template.

# Test scenario priorities are defined as very high, high, medium, low.

# Test Case?

# Explains how a scenario to be tested. Test cases are obtained from the requirement documents and written based on Black box test case design, Equivalence Class partitioning (ECP), Boundary Valve Analysis (BVA), Decision Table, Error Guessing, State transition.

# Test case priorities are defined as p0, p1, p2, p3.

# Test Strategies?

# 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.

# Various sections in Test Strategies:

1. Scope and overview
2. Test Approach
3. Testing tools
4. Industry standards to follow
5. Test deliverables
6. Testing metrics
7. Requirement Traceability Matrix
8. Risk and mitigation
9. Reporting tool
10. Test summary

# Test Plan?

# Test plan document is also shared with the stakeholders. The stakeholders get to know the scope, approach, objectives, and schedule of software testing to be done

1. Test Plan Identifier: Example: ProjectName\_0001
2. References: This section is to specify all the list of documents that support the test plan which you are currently creating.
3. Introduction:  purpose and scope of the project
4. Test Items: List of test items that needed to be tested.
5. Features to be Tested: List of all features that are to be tested. (Login/dashboard/reports)
6. Features Not to Be Tested: features not included in testing
7. Approach
8. Pass/Fail Criteria
9. Suspension Criteria
10. Test Deliverables
11. Testing Tasks
12. Environmental Needs
13. Responsibilities
14. Staffing and Training Needs
15. Schedule
16. Risks and Contingencies
17. Approvals

# STLC cycle?

# Software Testing Life Cycle (STLC)

# STLC - Software Testing Life CycleR

# Requirement Analysis: BRS (Business Requirement Specifications) is the entry criteria. Tester team should study BRS document.

# Test Planning: Test manager or test lead determine the cost, time, resources planning, roles and responsibilities, tools selection, training requirements etc.

# Test Design: Test team prepares test cases, test scripts and test data.

# Test Environment Setup: This is done based on hardware and software requirement list. Smoke test is also done in this phase.

# Test Execution: Testing executes the test cases, if pass or fail, test case should be updated. Defects should report through bug tracking tool.

# Test Closure: Test closure reports and test matrix reports are prepared. Test team will evaluate the cycle by arranging meetings.

# Bug life cycle/defect life cycle?

# Bug Life Cycle Defect Life Cycle

# Test Case Template with explanation?

# ****PROJECT NAME:**** Name of the project the test cases belong to. ****MODULE NAME:**** Name of the module the test cases belong to. ****REFERENCE DOCUMENT:****Mention the path of the reference documents (if any such as Requirement Document, [Test Plan](https://www.softwaretestingmaterial.com/test-plan/), Test Scenarios etc.,). ****CREATED BY:**** Name of the Tester who created the test cases. ****DATE OF CREATION:**** When the test cases were created. ****REVIEWED BY:****Name of the Tester who created the test cases. ****DATE OF REVIEW:****When the test cases were reviewed. ****EXECUTED BY:**** Name of the Tester who executed the test case ****DATE OF EXECUTION:**** When the test case was executed ****TEST CASE ID:**** Each test case should be represented by a unique ID. It’s good practice to follow some naming convention for better understanding and discrimination purpose. ****TEST SCENARIO:**** Test Scenario ID or title of the test scenario. ****TEST CASE:**** Title of the test case. ****PRE-CONDITION:**** Conditions which needs to meet before executing the test case. ****TEST STEPS:**** Mention all the test steps in detail and in the order how it could be executed. ****TEST DATA:**** The data which could be used an input for the test cases. ****EXPECTED RESULT:**** The result which we expect once the test cases were executed. It might be anything such as Home Page, Relevant screen, Error message etc., ****POST-CONDITION:**** Conditions which needs to achieve when the test case was successfully executed. ****ACTUAL RESULT:**** The result which system shows once the test case was executed. ****STATUS:**** If the actual and expected results are same, mention it as Passed. Else make it as Failed. If a test fails, it has to go through the bug life cycle to be fixed.

# Bug Reporting Template?

Defect ID Add a Defect ID using a naming convention followed by your team. The Defect ID will be generated automatically in case of defect management tool.

Reporter Name: Name of the one who found the defect (Usually tester’s name but sometimes it might be Developer, Business Analyst, Subject Matter Expert (SME), Customer)

Defect Reported Date: Date of the defect reported

Who Detected: Add the designation of the one who found the defect. E.g. QA, Developer, Business Analyst, SME, Customer

How Detected: Testing, Review, Walkthrough

Project Name: Add name of the project. (If it’s a product, add product name)

Release/Build Version: Add the build version details here

Defect/Enhancement: Add whether it is defect or improvement

Environment: Add Operation Systems details, Browser Details and any other related to the test environment.

(E.g. Windows 8/Chrome 48.0.2564.103)

Priority: Add the priority of the bug

(E.g. High/Medium/Low)

Severity: Add the severity of the bug

(E.g. Critical/High/Medium/Low)

Status: Add the status of the bug. If you just found and posting it then it will be New. The status of the bug will change.

(E.g. New/ Assigned/ Open/ Fixed/ Test/ Verified/ Closed/ Reopen/ Duplicate/ Deferred/ Rejected/ cannot be fixed/ Not Reproducible/ Need more information)

Description: Add a detailed description.

Steps to reproduce: Mention steps in detail. So that even the one who has no idea about the application also could reproduce the bug.

URL: Add the URL of the application (If available)

Expected Result: Mention the expected result here which is available in your test case document.

Actual Result: Mention the actual result here which is available in your test case document.

# Defect Close Date: Add the defect close date only once you ensure that the defect is not reproducible.

# Test Metrics?

# It is used to monitor and control process and product.

# Process Metrics- Used in test preparation and execution.

# Product Metrics- used in defect analysis phase of STLC.

# Requirement traceability Metrics?

# It is used to trace the requirements to the test that are needed to verify whether requirements are fulfilled.

# Types: Forward traceability- Mapping requirements to test cases

# Backward Traceability- Mapping test cases to requirements.

# Test Deliverables?

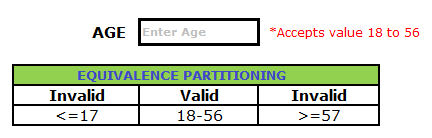
# 1. Test Strategy 2. [Test Plan](https://www.softwaretestingmaterial.com/test-plan/) 3. Effort Estimation Report 4. Test Scenarios 5. [Test Cases](https://www.softwaretestingmaterial.com/test-case-template-with-explanation/)/Scripts 6. Test Data 7. [Requirement Traceability Matrix (RTM)](https://www.softwaretestingmaterial.com/requirements-traceability-matrix/) 8. [Defect Report/Bug Report](https://www.softwaretestingmaterial.com/bug-report-template/) 9. Test Execution Report 10. Graphs and [Metrics](https://www.softwaretestingmaterial.com/test-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 clients.

# Equivalence Partitioning Testing Techniques?

# Equivalence Partitioning is also known as Equivalence Class Partitioning. In equivalence partitioning, inputs to the software or system are divided into groups that are expected to exhibit similar behavior, so they are likely to be proposed in the same way. Hence selecting one input from each group to design the test cases. Equivalence Partitioning is also known as Equivalence Class Partitioning. In equivalence partitioning, inputs to the software or system are divided into groups that are expected to exhibit similar behavior, so they are likely to be proposed in the same way. Hence selecting one input from each group to design the test cases.

**Example 1:**

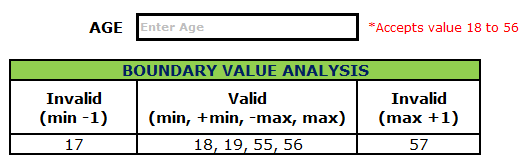
Assume, we have to test a field which accepts Age 18 – 56

[](https://i0.wp.com/www.softwaretestingmaterial.com/wp-content/uploads/2016/03/Equivalence-Partitioning-1.png?ssl=1)

**Boundary value analysis?**

Boundary value analysis (BVA) is based on testing the boundary values of valid and invalid partitions.

Assume, we must test a field which accepts Age 18 – 56

[](https://i1.wp.com/www.softwaretestingmaterial.com/wp-content/uploads/2016/03/Boundary-Value-Analysis-1.png?ssl=1)

Minimum boundary value is 18

Maximum boundary value is 56

# Decision Table Test Case?

This test technique is appropriate for functionalities which has logical relationships between inputs

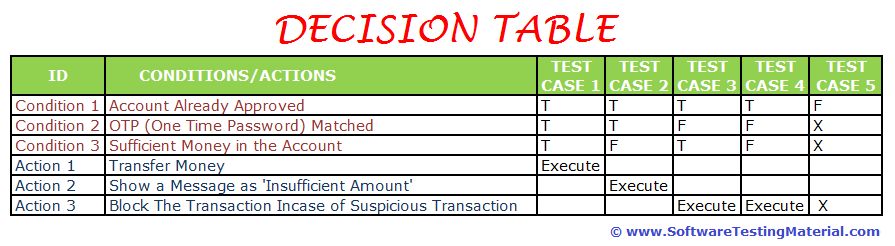
### Examples on Decision Table Test Case Design Technique:

Take an example of transferring money online to an account which is already added and approved.

Here the conditions to transfer money are account already approved, otp (one-time password) matched, sufficient money in the account.

And the actions performed are transfer money, show a message as insufficient amount, block the transaction in case of suspicious transaction.

Here we decide under what condition the action be performed Now let’s see the tabular column below.

[](https://i0.wp.com/www.softwaretestingmaterial.com/wp-content/uploads/2016/03/Decision-Table.png?ssl=1)

### ****What is Defect Priority?****

Defect priority can be defined as how soon the defect should be fixed. It gives the order in which a defect should be resolved. Developers decide which defect they should take up next based on the priority. It can be High, Medium or Low.

### ****What is Defect Severity?****

Bug/Defect severity can be defined as the impact of the bug on customer’s business. It can be Critical, Major or Minor. In simple words, how much effect will be there on the system because of a defect.

**What is QA Environment?**

A QA environment is a place where we implement our upgraded procedure on data, hardware and software to test the resulting Wave set Application.

**What is Staging Environment?**

Staging environment is required by the QA team to determine how an application performs in production.

**What is Defect Triage?**

Defect Triage is a process of dividing the open bugs in a software application into different categories to analyze the defects and finding the required actions to be taken.

**What is Defect Age?**

Defect age can be defined as the time interval between the ‘Date of Defect detection’ and ‘Date of Defect Closure’.

**What is Error Seeding?**

Error Seeding is a process of adding known errors intendedly in a program to identify the rate of error detection. These helps in the process of estimating the testers skills of finding bugs and know the ability of the application.

# 