Manual Testing

Phases of Manual Testing

- Testing Concepts - (Theory)
- Testing Process - (Theory + Practical)
- Agile Process - (Theory)
- Live Project - (Practical)

Testing Concepts

What is Software ?

- A Software is a collection of computer programs that helps us to perform tasks.

- Types of Software:
  - System Software
    - Ex. Device Drivers, OS, Servers, Utilities etc.
  - Programming Software
    - Ex. Compilers, Debuggers, Interpreters etc.
  - Application Software
    - Ex. Industrial Automation, Business Software, Games, Telecoms etc.

Product Vs Project

- If software application is developed for specific customer requirement then it is called **Project.**

- If software application is developed for multiple customers requirement then it is called **Product.**

What is Software Testing

- **Software Testing** is a part of software development process.
- **Software Testing** is an activity to detect and identify the defects in the software.
- The **objective of testing** is to release **quality product** to the client.

Why do we need testing?

- Ensure that software is bug free.
- Ensure that system meets customer requirements and software specification.
- Ensure that system meets end user expectations.
- Fixing the bugs identified after release is expensive.

Software Quality

- **Quality :** Quality is defined as justification of all the requirements of a customer in a product.
- Quality Software is reasonably
    - Bug - free
    - Delivered on Time
    - Within budget
    - Meets requirements and/or expectations
    - Maintainable

Error, bug & failure

- **Error :** Any incorrect human action that produces a problem in the system is called an Error.
- **Defect/Bug :** Deviation from the expected behavior to the actual behavior of the system is called defect.
- **Failure :** The deviation identified by end-user while using the system is called a failure.

Why there are bugs in software?

- Miscommunication or no communication
- Software Complexity
- Programming Errors
- Changing requirements
- Lack of skilled testers etc.

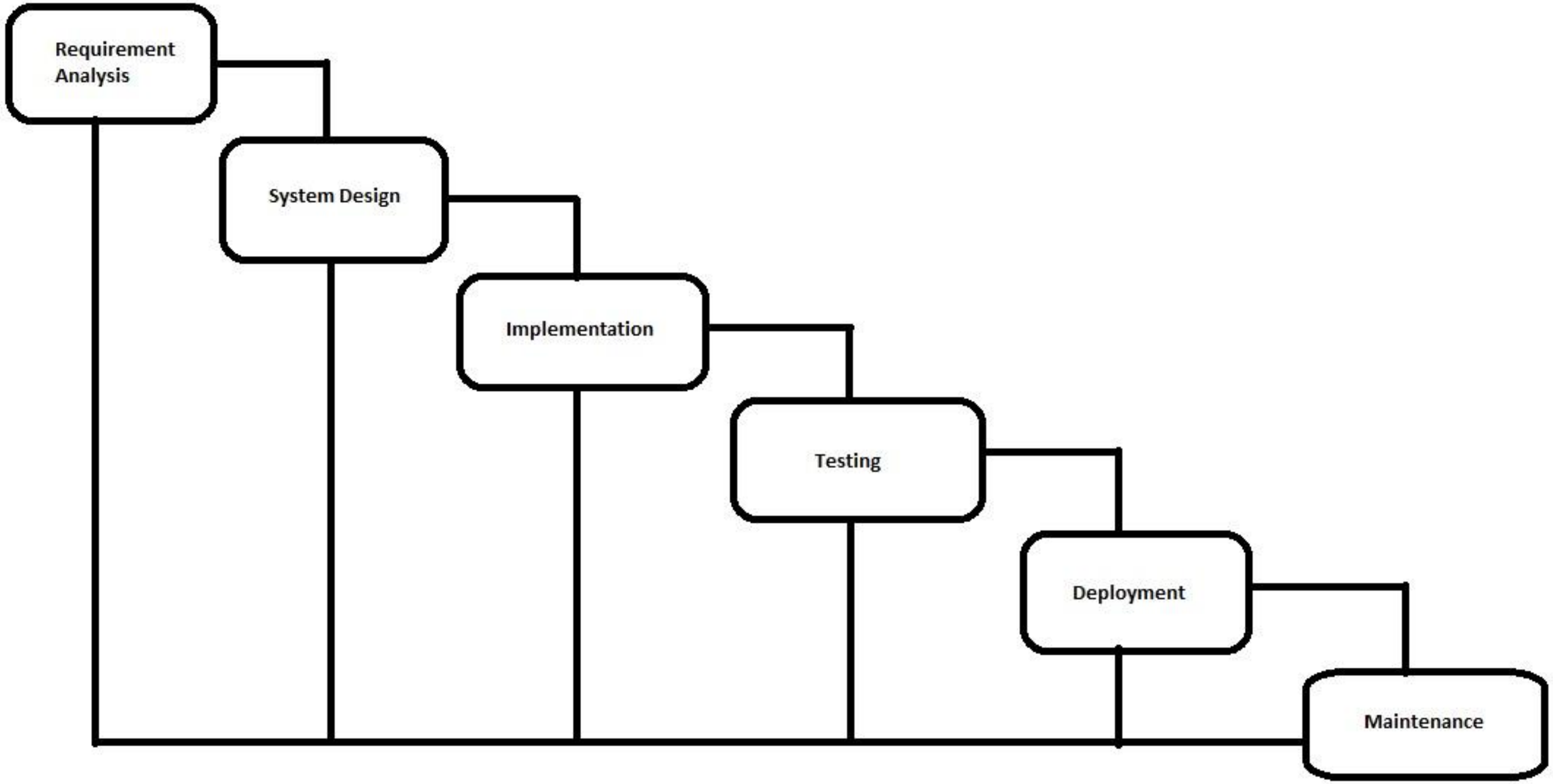
Software Development Life Cycle (SDLC)

- SDLC, Software Development Life Cycle is a process used by software industry to design, develop and test high quality softwares.
- The SDLC aims to produce a high quality software that meets customer expectations.

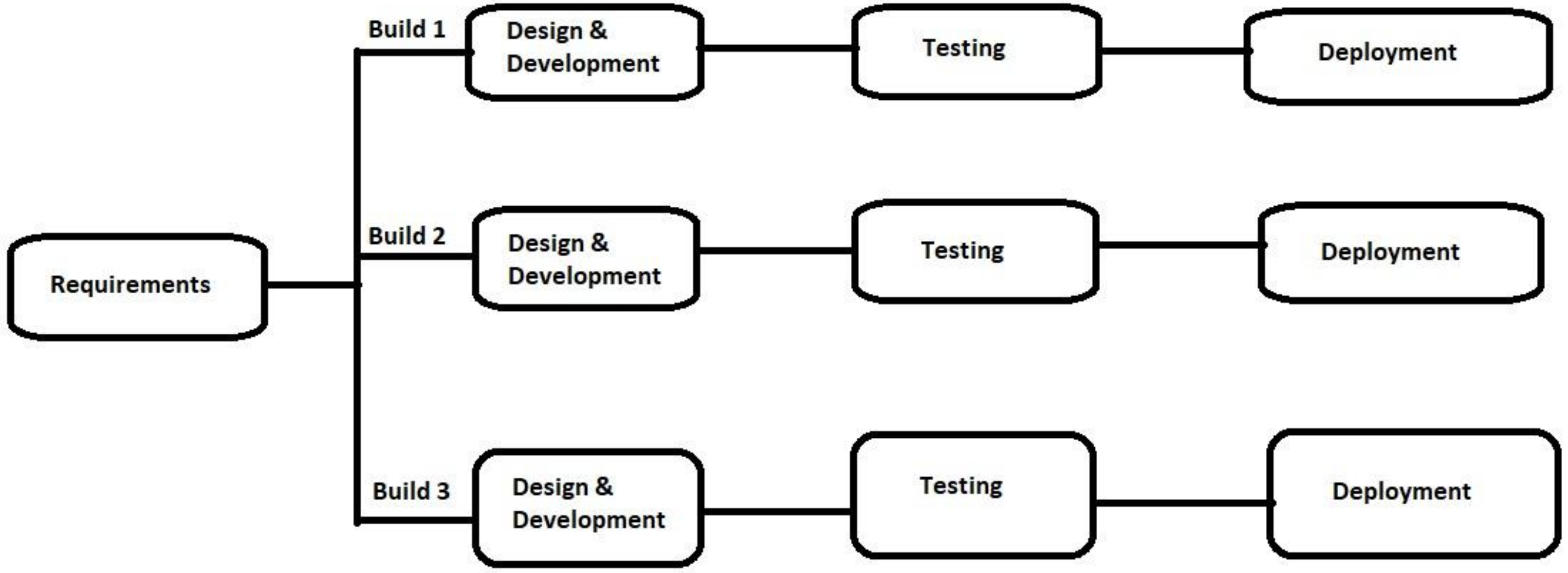
SDLC Models

- Waterfall Model
- Incremental Model
- Spiral Model
- V Model
- Agile Model

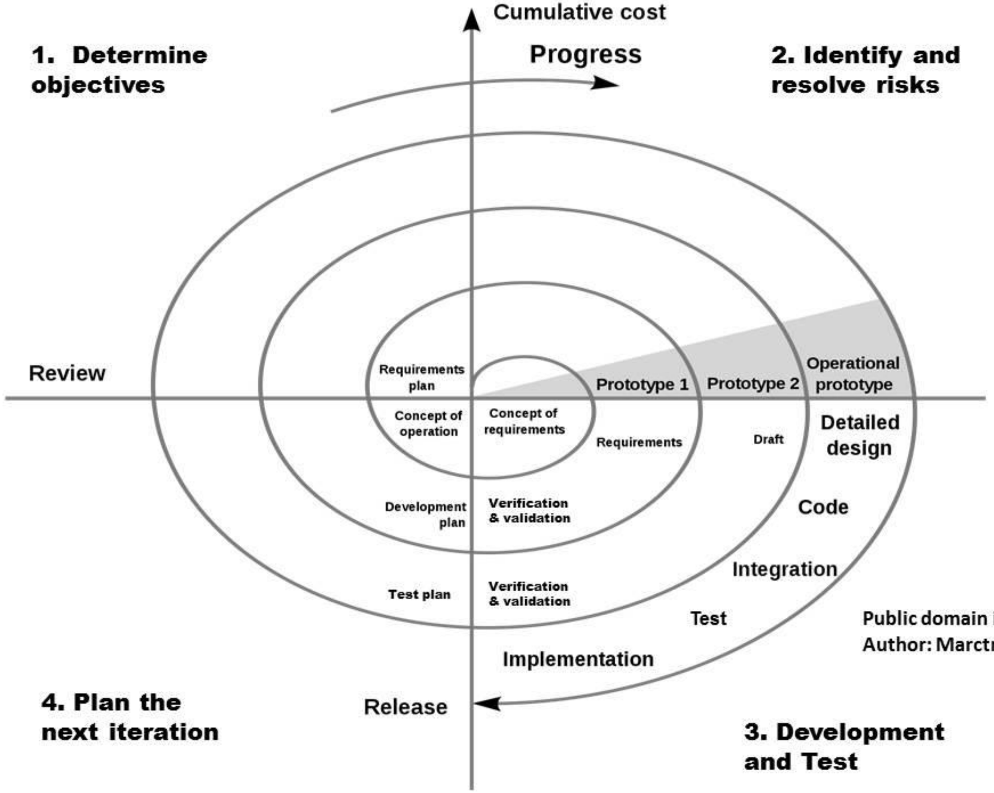
Waterfall Model



Incremental/ Iterative Model

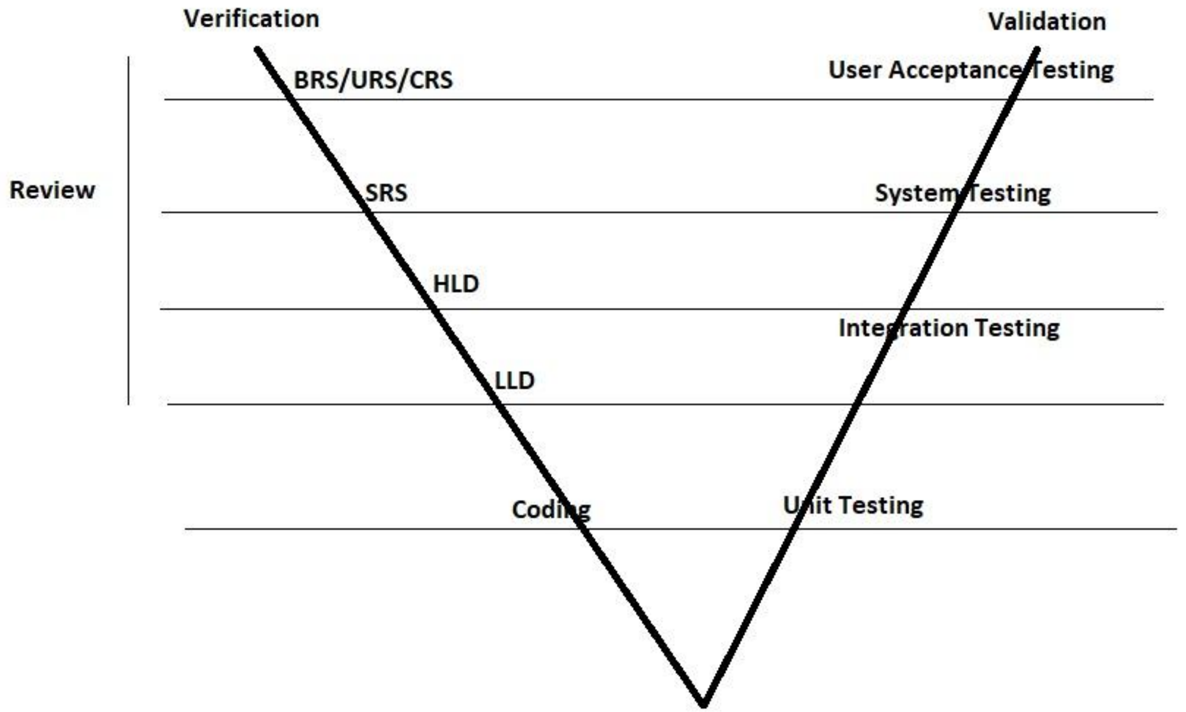


Spiral Model

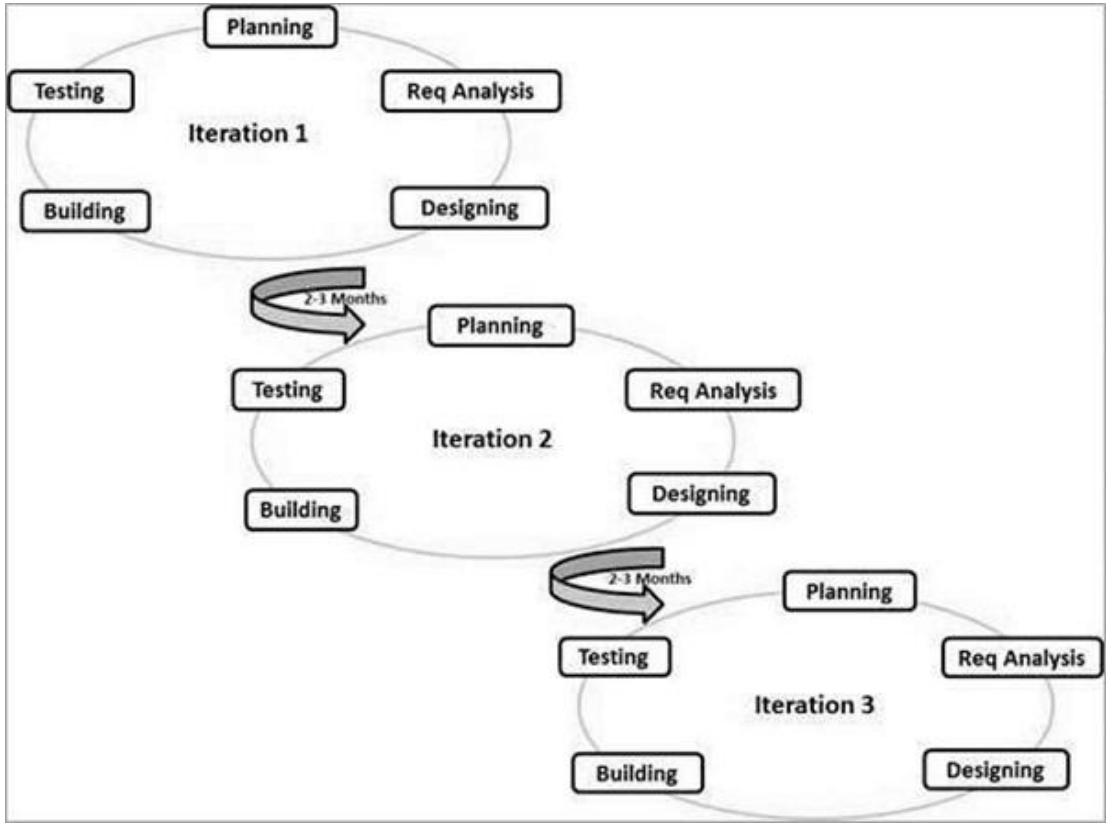


Public domain image
Author: Marctroy

V Model


Verification
Validation
BRS/URS/CRS
User Acceptance Testing
Review
SRS
System Testing
HLD
Integration Testing
LLD
Coding
Unit Testing

Agile Model


Planning
Req Analysis
Designing
Building
Testing
Iteration 1
2-3 Months
Planning
Req Analysis
Designing
Building
Testing
Iteration 2
2-3 Months
Planning
Req Analysis
Designing
Building
Testing
Iteration 3

Verification V/S Validation

- Verification checks weather we are building the right system.
- It typically involves :
  - Reviews
  - Walkthroughs
  - Inspections
- Validation checks whether we are building the system right.
- Takes place after verifications are completed.
- Validation typically involves actual testing.
  - System testing

Static V/S Dynamic Testing

- Static Testing is an approach to test project documents in the form of Reviews, Walkthroughs and Inspections.

- Dynamic Testing is an approach to test the actual software by giving inputs and observing results.

Review

- Conducts on documents to ensure correctness and completeness.
- Examples:
  - Requirement Reviews
  - Design Reviews
  - Code Reviews
  - Test Plan Reviews
  - Test Case Reviews

Walkthrough

- **It is a formal review and we can discuss/raise the issues at peer level.**
- **Also walkthrough does not have minutes of the meet. It can happen at any time and conclude just like that no schedule as such.**

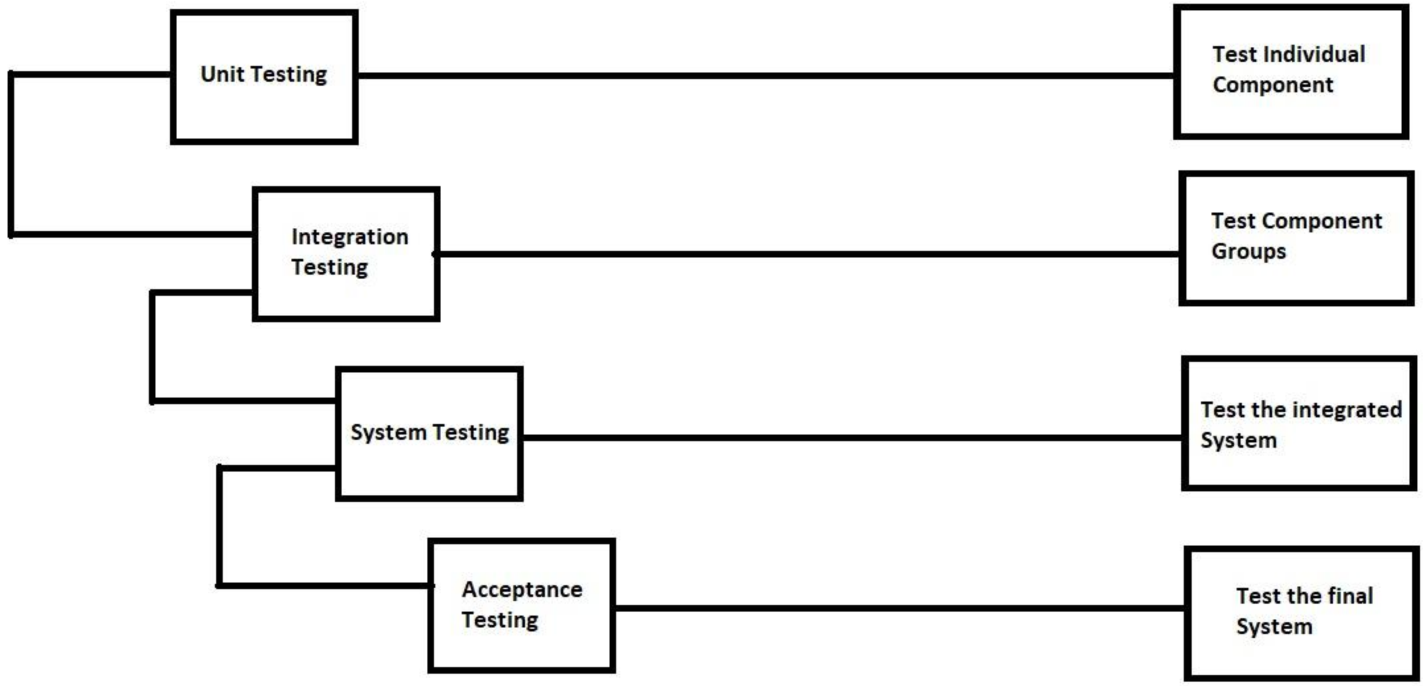
Inspection

- It is a formal approach to the requirements schedule.
- At least 3-8 people will sit in the meeting 1- reader 2- writer 3 -moderator plus concerned.
- Inspection will have a proper schedule which will be intimated via email to the concerned developer/ tester.

Levels of Software Testing

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

Levels of Software Testing



Unit Testing

- **A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output.**
- **Unit testing conducts on a single program or single module.**
- **Unit testing is white box testing techinque.**
- **Unit testing is conducted by developer.**
- **Unit Testing techniques :**
    - **Basis path testing**
    - **Control Structure Testing**
        - **Conditional Coverage**
        - **Loop Coverage**
    - **Mutation Testing**

Integration Testing

- **In Integration Testing, individual software modules are integrated logically and tested as group.**
- **Integration Testing focuses on checking data communication amongst these modules.**
- **Integration Testing is white box testing technique.**
- **Integration Testing is conducted by the developers.**
- **Approaches :**
  - **Top Down Approach**
  - **Bottom Up Approach**
  - **Sandwich Approach (Hybrid)**

- **Stub : Is called by the Module under test.**

- **Driver: Calls the Module to be tested.**

Bottom-Up Integration

- In the bottom up strategy, each module at lower level is tested with higher modules until all modules are tested. It takes help of Drivers for testing.

Top Down Integration

- **In Top to Down approach, testing takes place from top to down following the control flow of the software system. Takes help of stubs for testing.**

System Testing

- Testing overall functionality of the application with respective client requirements.
- It is a black box testing technique.
- This testing is conducted by testing team.
- Before conducting system testing we should know the requirements.
- System Testing focuses on below aspects.
    - Graphical User Interface Testing (GUI)
    - Functional Testing
    - Non-functional Testing
    - Usability Testing

User Acceptance Testing

- **After completion of system testing UAT team conducts acceptance testing in two levels.**
    - **Alpha Testing**
    - **Beta Testing**

Testing Methodologies

- White box Testing
- Black box Testing
- Grey box Testing

White Box Testing

- White Box Testing conducts on internal logic of the system.
- Programming skills are required.
- Ex. Unit Testing & Integration Testing

Black Box Testing

- **Testing conducts on functionality of the application whether it is working according to customer requirements or not.**
- **Programming skills are not required.**
- **Ex. System Testing & UAT Testing**

Grey Box Testing

- Both combination of white box and black box testing.
- Ex. Database Testing

Black Box Testing Types

- GUI Testing
- Usability Testing
- Functional Testing
- Non-functional Testing

What is GUI?

- There are two types of interfaces in a computer application.
    - Command Line Interface is where you type text and computer responds to that command.
    - GUI stands for Graphical User Interface where you interact with the computer using computer using images rather than text.

GUI Testing

- Graphical User Interface (GUI) testing is the process of testing the system's GUI.
- GUI Testing involves checking the screens with the controls like menu, buttons, icons and all types of bars - tool bar, menu bar, dialog boxes and windows etc.
- During GUI Testing Test Engineers validates user interface of the application as following aspects:
    - Look and Feel
    - Easy to use
    - Navigations & Shortcut Keys
- GUI Objects :
    - Window, Dialog Box, Push Button, Radio Button, Radio Group, Tool Bar, Edit Bo Text box, Check Box, List Box, Drop Down Box, Combo box, Tab, Tree view, Progress bar, Table, Scroll Bar Etc.

Checklist For GUI Testing

- It checks if all the basic elements are available in the page or not.
- It checks the spelling of the objects.
- It checks the alignments of the objects.
- It checks the content display in web pages.
- It checks if the mandatory fields are highlights or not.
- It checks consistency in background color and color font type and font size etc.

Usability Testing

- **During this testing validates application provided context sensitive help or not to the user.**

- **Checks how easily the end user are able to understand and operate the application is called usability testing.**

Functional Testing

- Object Properties Coverage
- Input Domain Coverage (BVA , ECP)
- Database Testing/ Backend Coverage
- Error Handling Coverage
- Calculations/ Manipulations Coverage
- Links Existence & Links Execution
- Cookies and Sessions

Object Properties Testing

- **Every Object has certain properties.**
  - **Ex. Enable, Disable, Focus, Text etc.**
- **During Functional Testing Test Engineers Validate properties of objects in run time.**

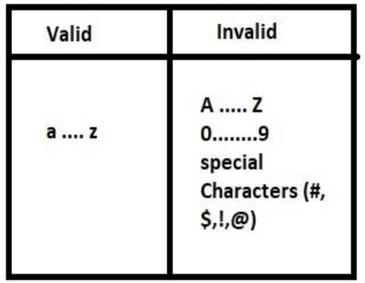
Input Domain Testing

- During input domain testing Test Engineers validate data provided to the application w.r.t. Value and length
- There are two techniques in Input Domain Techniques :
    - Equivalence Class Partition (ECP) - Will check value
    - Boundary Value Analysis (BVA) - Will check length

ECP & BVA

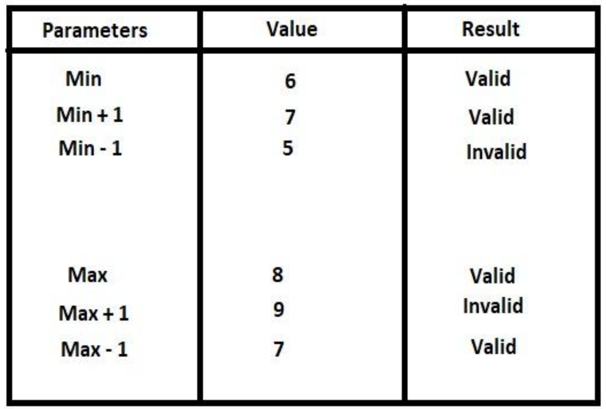
- **Username field allows only lower case with min 6 max and 8 letter**

ECP for Username



Valid	Invalid
a .... z	A ..... Z 0........9 special Characters (#, $,!,@)

BVA for Username



Parameters	Value	Result
Min	6	Valid
Min + 1	7	Valid
Min - 1	5	Invalid
Max	8	Valid
Max + 1	9	Invalid
Max - 1	7	Valid

DatabaseTesting

- During Database Testing Test Engineers validate the data w.r.t. database
- Validates DML operations (Insert, Update, Delete & Select)
- SQL Language : DDL, DML, DCL etc.
    - DDL - Data Definition Language - Create, alter , drop
    - DML - Data Manipulation Language - Insert, Update, Select , Delete
    - DCL - Commit, roll back etc.

Error Handling Testing

- Validate error messages thrown by the application when we provide invalid data.
- The error messages should be clear and easy to understand to the user.

Calculations/ Manipulations Testing

- **Validate mathematical calculations.**

Links Coverage

- **Links existence - Links placed in the appropriate location or not.**
- **Links execution - Link is navigating to appropriate page or not.**
- **Types of links :**
    - **Internal links**
    - **External links**
    - **Broken links**

Cookies & Sessions

- **Cookie - Temporary internet files which are created at client side when we open the web sites. These files contains User data.**
- **Session - Sessions are time slots which are allocated to the user at the server side.**

Non Functional Testing

- Performance Testing
    - Load Testing
    - Stress Testing
    - Volume Testing
- Security Testing
- Recovery Testing
- Compatibility Testing
- Configuration Testing
- Installation Testing
- Sanitation Testing

Performance Testing

- Load - Testing speed of the system while increasing the load gradually till the customer expected number.
- Stress - Testing speed of the system while increasing/ reducing the load on the system to check anywhere its breaking.
- Volume - Check how much volumes of data is able to handle by the system.

Security Testing

- Testing security provided by the system.
- Types :
    - Authentication
    - Access Control / Authorization
    - Encryption / Decryption

Recovery Testing

- Testing Recovery provided by the system. Whether the system recovering abnormal to normal condition or not.

Compatibility Testing

- Testing Compatibility of the system w.r.t OS, H/W & Browsers
    - Operating System Compatibility (Linux, IOS, Windows)
    - Hardware Compatibility (Configuration Testing)
    - Browser Compatibility
    - Forward & Backward Compatibility

Installation Testing

- **Testing Installation of the application on Customer expected platforms and check installation steps, navigation, how much space is occupied in memory.**
- **Check uninstallation.**

Sanitation / Garbage Testing

- **Check whether application is providing any extra/ additional features beyond the customer requirements.**

Testing Process / STLC

Testing Terminology

- **Adhoc Testing:**
  - **Software Testing performed without proper planning and documentation.**
  - **Testing is carried out with the knowledge of the tester about the application and the tester tests randomly without following the specifications/ requirements.**

- **Monkey Testing:**
  - **Tests the functionality randomly without knowledge of application and test cases is called monkey testing.**

Testing Terminology Cont...

- **Re- Testing:**
  - Testing functionality repetitively is called Re-testing.
  - Re- Testing gets introduced in the following two scenarios :
    - Testing functionality with multiple inputs to confirm the business validations are implement or not.
    - Testing functionality in the modified build is to confirm the bug fixers are made correctly or not.
- **Regression Testing**
  - It is a process of identifying various features in the modified build where there is a chance of getting side-effects and retesting these features.
  - The new functionalities added to the existing system modifications made to the existing system.
  - It must be noted that a bug fixer might introduce side-effects and a regression testing is helpful to identify these side effects.

Testing Terminology Cont...

- **Sanity Testing:**
    - **This is a basic functional testing conducted by test engineer whenever receive build from development team.**

- **Smoke Testing:**
    - **This is also basic functional testing conducted by developer or tester before releasing the build to the next cycle.**

Testing Terminology Cont...

- **End to End Testing:**
  - **Testing the overall functionalities of the system including the data integration testing among all the modules is called end-to-end testing.**

- **Exploratory Testing:**
  - **Exploring the application and understanding the functionalities adding or modifying the existing test cases for better testing is called exploratory testing.**

Testing Terminology Cont...

- **Globalization Testing / Internationalization Testing:**
    - **Checks if the application has a provision of setting and changing languages data and time format and currency etc. If it is designed for global users, it is called globalization testing.**

- **Localization Testing:**
    - **Checks default languages currency date and time format etc. If it is designed for a particular locality of users is called Localization testing.**

Testing Terminology Cont...

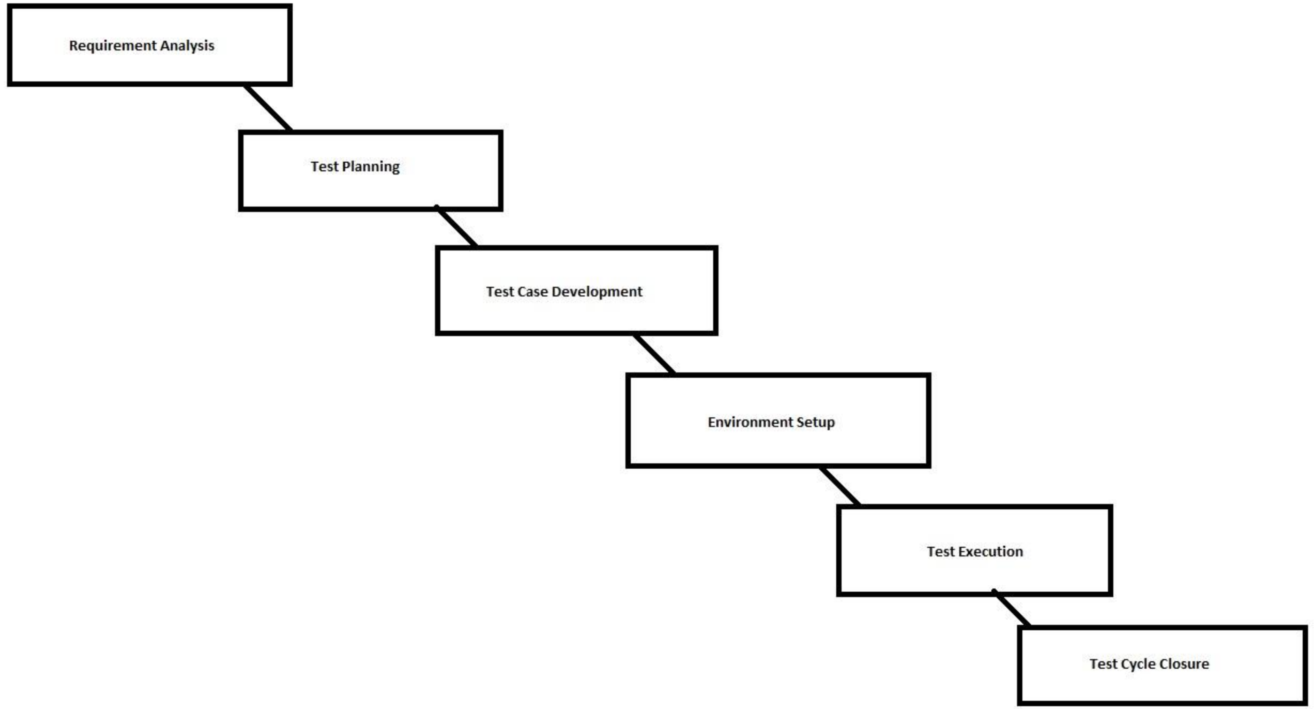
- **Positive Testing (+ve):**
  - Testing conducted on the application in a positive approach to determine what system is supposed to do is called a positive testing.
  - Positive testing helps in checking if the customer requirements are justifying the application or not.

- **Negative Testing (-Ve):**
  - Testing a software application with a negative perception to check what system is not supposed to do is called negative testing.

Software Testing Life Cycle (STLC)

- **Requirement Analysis**
- **Test Planning**
- **Test Designing**
- **Test Execution**
- **Test Closure**

Software Testing Life Cycle (STLC)



Test Plan Definition

- **A document describing the scope, approach, resources and schedule of testing activities.**

- **It identifies test items, the feature to be tested, the testing tasks, who will do each task and any risks requiring contingency planning.**

- **A detail of how the test will proceed, who will do the testing, what will be tested, in how much time the test will take place, and to what quality level the test will be performed.**

Test Plan Contents

- Objective
- Features to be tested
- Features not to be tested
- Test Approach / Strategy
- Entry Criteria
- Exit Criteria
- Suspension & Resumption Criteria
- Test Environment
- Test Deliverables & Milestones
- Schedules
- Risks & Mitigations
- Approvals

Use case, Test Scenario & Test Case

- Use Case:
  - Use case describes the requirement.
  - Use case contains THREE Items.
    - Actor
    - Action/Flow
    - Outcome
- Scenario :
  - Possible area to be tested (What to test)
- Test Case :
  - How to test
  - Describes test steps, expected results & actual result.

Use Case V/s Test Case

- **Use Case: - Describes functional requirement, prepared by Business Analyst(BA) / Product Owner**
- **Test Case: - Describes Test Steps/ Procedure, prepared by Test Engineer**

Test Case V/s Test Scenario

- **Test case consist of set of input values, execution precondition, expected results and executed post condition, developed to cover certain test condition. While Test scenario is nothing but test procedure.**
- **Test cases are derived (or written) from test scenario. The scenarios are derived from use cases. The scenarios are derived from use cases.**
- **In short, Test Scenario is “What to be tested” and Test Case is “How to be tested.”**
    - **Example:**
    - **Test Scenario : Checking the functionality of Login**
        - **TC1 : Click the button without entering username and password.**
        - **TC2 : Click the button only entering User name.**
        - **TC3 : Click the button while entering wrong username and wrong password.**

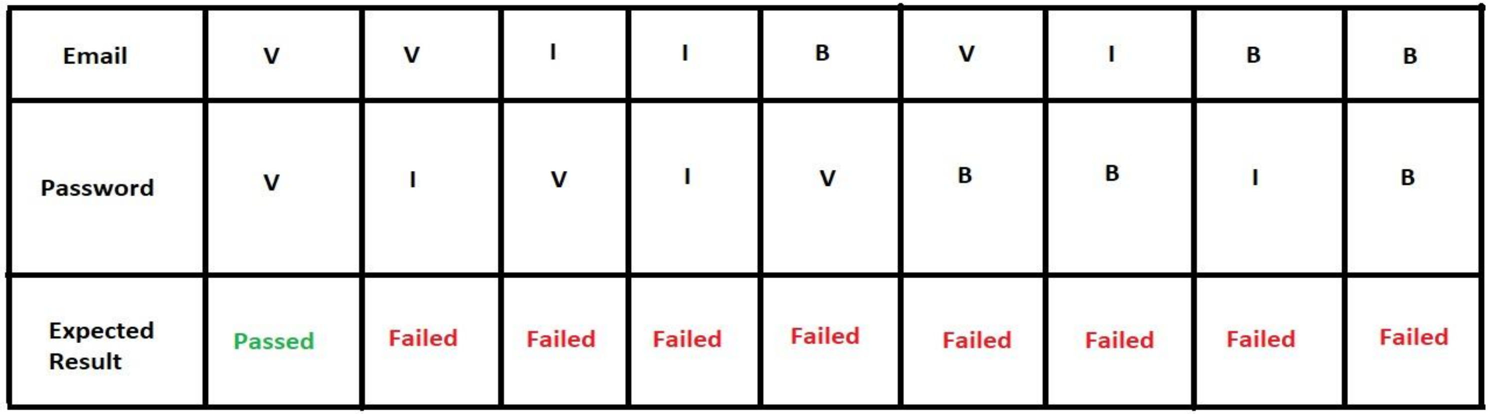
Positive V/s Negative Test Cases

- Requirement :
  - For example, if a textbox is listed as a feature and in SRS it is mentioned as Text box accep
    6 - 20 characters and only alphabets.
- Positive Test Cases :
  - Textbox accepts 6 characters
  - Textbox accepts up to 20 chars length.
  - Textbox accepts any value between 6-20 chars length.
  - Textbox accepts all alphabets.
- Negative Test Cases :
  - Textbox should not accepts less than 6 chars
  - Textbox should not accepts chars more than 20 chars.
  - Textbox should not accept special characters.
  - Textbox should not accept numerical.

Test Case Design Techniques

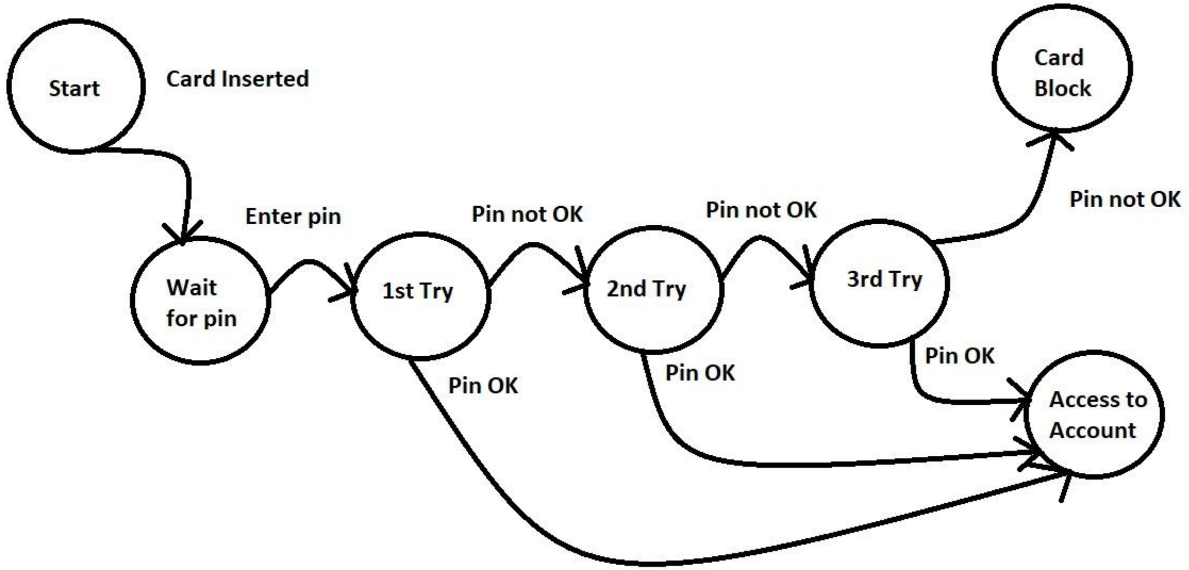
- **Boundary Value Analysis (BVA) - Range of values**
- **Equivalence Class Partitioning (ECP) - Valid / Invalid**
- **Decision Table Testing**
- **State Transition Diagram**
- **Use Case Testing**

Decision Table

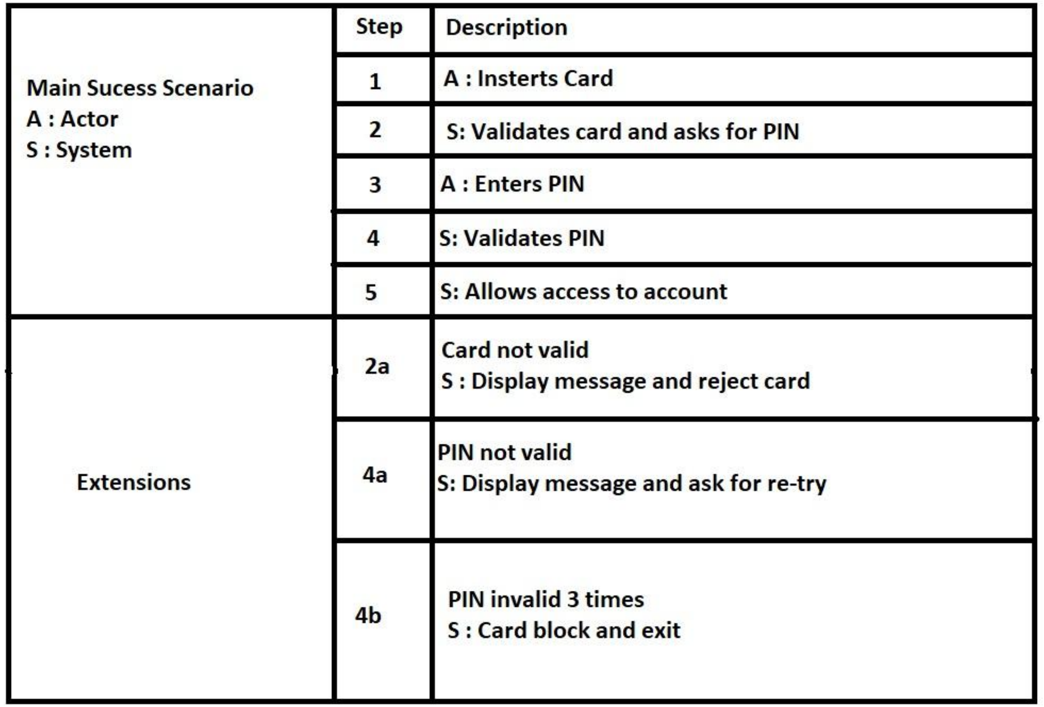


Email	V	V	I	I	B	V	I	B	B
Password	V	I	V	I	V	B	B	I	B
Expected Result	Passed	Failed	Failed	Failed	Failed	Failed	Failed	Failed	Failed

State Transition Diagram



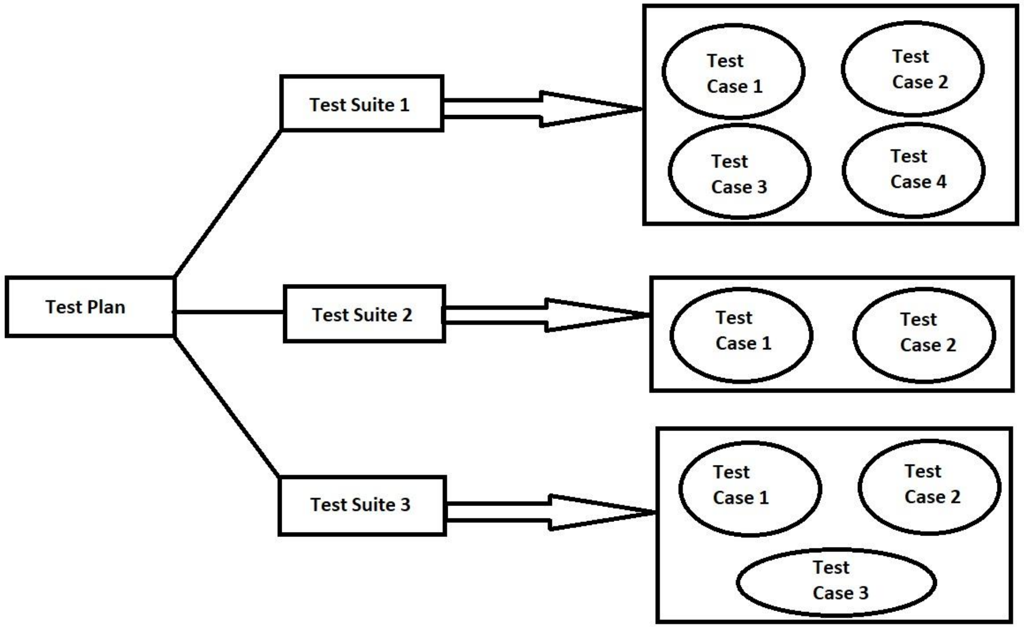
Use Case



	Step	Description
Main Sucess Scenario A : Actor S : System	1	A : Insterts Card
	2	S: Validates card and asks for PIN
	3	A : Enters PIN
	4	S: Validates PIN
	5	S: Allows access to account
Extensions	2a	Card not valid S : Display message and reject card
	4a	PIN not valid S: Display message and ask for re-try
	4b	PIN invalid 3 times S : Card block and exit

Test Suite

- **Test Suite is group of test cases which belongs to same category.**

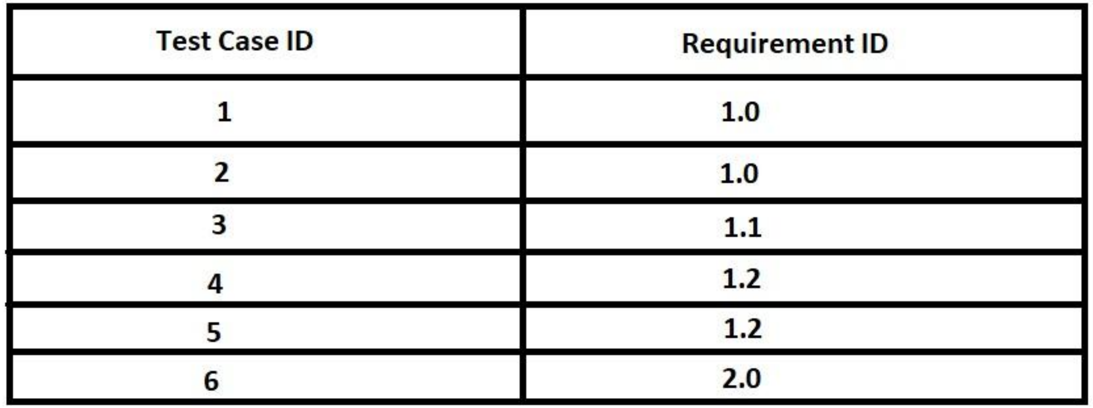


Test Case Contents

- **Test Case ID**
- **Test Case Title**
- **Description**
- **Pre-Condition**
- **Priority (P0, P1, P2, P3) - Order**
- **Requirement ID**
- **Steps/ Actions**
- **Expected Result**
- **Actual Result**
- **Test Data**

Requirement Traceability Matrix (RTM)

- **RTM - Requirement Traceability Matrix**
- **Used for mapping of Requirements w.r.t Test Cases**



Test Case ID	Requirement ID
1	1.0
2	1.0
3	1.1
4	1.2
5	1.2
6	2.0

Characteristics of Good Test Cases

- **Should be accurate and tests what it is intended to test.**
- **No unnecessary steps should be included in it.**
- **It should be reusable.**
- **It should be traceable to requirements.**
- **It should be compliant to regulations.**
- **It should be independent i.e. You should be able to execute it in any order without any dependency on other test cases.**
- **It should be simple and clear, any tester should be able to understand it by reading once.**

Test Data

- Test Data is required for Test Cases.
- Test Data will be provided as input to the cases.
- Prepared by Test Engineers.
- Maintains in Excel Sheets.

Test Environment Setup (Test Bed)

- **Test environment decides the software and hardware conditions under which a work product is tested.**
- **Activites :**
  - **Understand the required architecture, environment setup and prepare hardware and softwa requirement list for the Test Environment.**
  - **Setup test Environment and Test Data.**
  - **Perform smoke test on build**
- **Deliverables :**
  - **Environment ready with test data set up**
  - **Smoke test results.**

Test Execution

- During this phase test team will carry out the testing based on the test plans and the test cases prepared.

- Bugs will be reported back to the development team for correction and retesting will be performed.

Defect Reporting

- **Any mismatched functionality found in a application is called as Defect/ Bug/ Issue.**
- **During Test Execution Test Engineers are reporting mismatches as defects to developers through templates or using tools.**
- **Defect Reporting Tools :**
    - **Clear Quest**
    - **DevTrack**
    - **Jira**
    - **Quality Center**
    - **Bugzilla etc.**

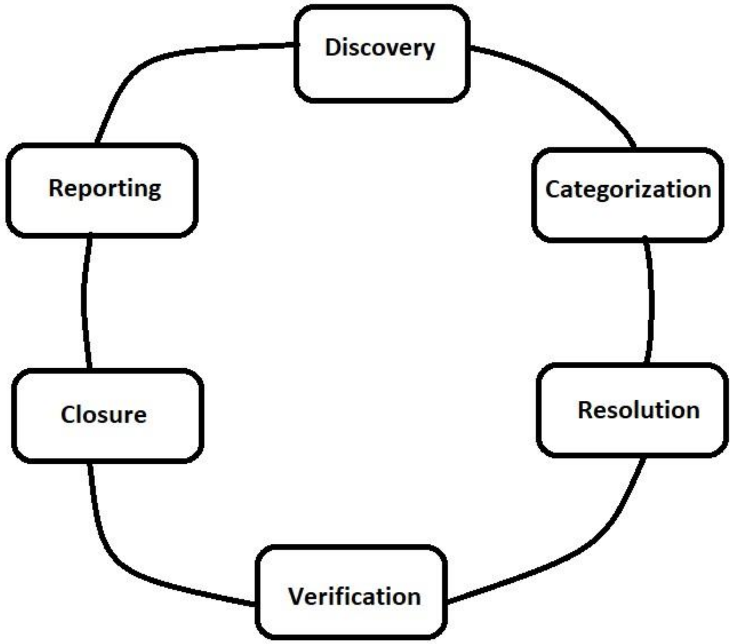
Defect Report Contents

- Defect_ID : Unique identification number for the defect.
- Defect Description : Detailed description of the defect including information about the module in which defect was found.
- Version : Version of the application in which defect was found.
- Steps : Detailed steps along with screenshots with which the developer can reproduce the defects.
- Date Raised : Date when the defect is raised.
- Reference : Where in you Provide Reference to the documents like. Requirement, design, architecture or may be even screenshots of the error to help understand the defect.
- Detected by : Name/ID of the tester who raised the defect
- Status : Status of the defect, more on this later.

Defect Report Contents Cont..

- Fixed By : Name/ ID of the developer who fixed it.
- Date Closed : Date when the defect is closed.
- Severity : Which describes the impact of the defect on the application.
- Priority : Which is related to defect fixing urgency. Severity Priority could be High / Medium / Low based on the impact urgency at which the defect should be fixed respectively.
- Defect Resolution :

Defect Management Process



Defect Severity

- Severity describes the seriousness of defect.
- In software testing, defect severity can be defined as the degree of impact a defect has on the development or operation of a component application being tested.
- Defect severity can be categorized into four class
  - Critical : This defect indicates complete shut-down of the process, nothing can proceed further.
  - High : It is a highly severe defect and collapse the system. However, certain parts of the system remain functional.
  - Medium : It cause some undesirable behavior, but the system is still functional
  - Low : It will not cause any major break-down of the system

Defect Priority

- Priority describes the importance of defect.
- Defect priority states the order in which a defect should be fixed. Higher the priority the sooner the defect should be resolved.
- Defect severity can be categorized into four class
  - High : The defect must be resolved as soon as possible as it is affects the system severely and can not be used until it is fixed.
  - Medium : During the normal course of the development activities defect should be resolved. I can wait until a new version is created.
  - Low : The defect is an irritant but repair can be done once the more serious defect have been fixed.

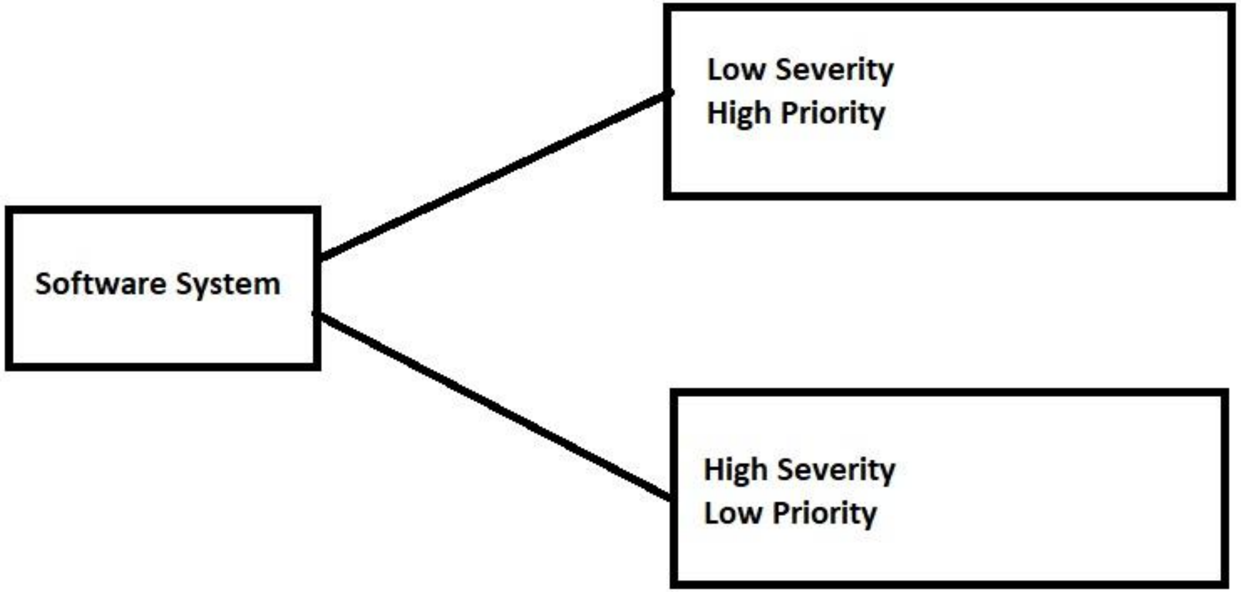
Tips for determining Severity of a defect

Severity

Priority

	Critical	Non - Critical
Urgent	Key feature does not work	Company logo in the wrong color
Low	Feature that is rarley used does not work.	The caption on an image is written in the wrong font.

A Software System can have a LS/HP & HS/LP



A Software System can have a LS/HP & HS/LP

- A very low severity with a high priority:
    - A logo error for any shipment website, can be of low severity as it is not going to affect the functionality of the website but can be of high priority as you do not want any further shipment to proceed with wrong logo.

- A very high severity with a low priority:

    - For flight operating website, defect in reservation functionality may be of high severity but c be a low priority as it can be scheduled to release in a next cycle.

More Examples...

- A very High Priority with a High Severity defect :
    - If ‘login’ is required for an application and the users are unable to login to the application with valid user credentials. Such defects need to be fixed with high importance. Since it is stopping the customer to progress further.

- A very Low Priority with a High Severity :
    - If an application crashes after multiple use of any functionality i.e. if ‘Save’ Button is used for 200 times and then the application crashes, such defects have High Severity because application gets crashed, but Low Priority because no need to debug right now you can debug it after some days.

More Examples...

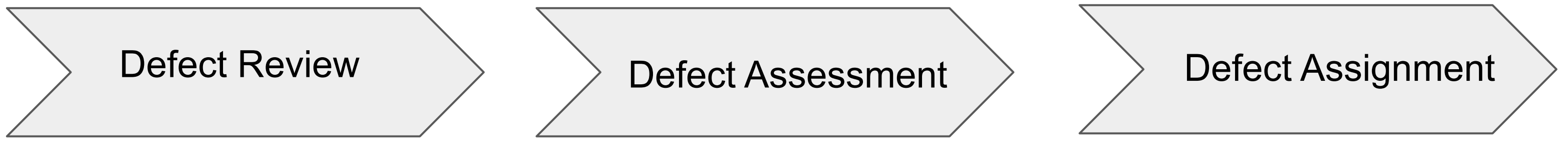
- A very High Priority with a Low Severity defect :
  - If in a web application, company name is misspelled or text “User Nam: ” is displayed instead of “Username:” on the application login page. In this case, Defect Severity is low as it is a spell mistake but Priority is high because of its high visibility.
- A very Low Priority with a Low Severity defect :
  - A spelling mistake in a page not frequently navigated by users.
- A very Low Priority with a High Severity defect :
  - Application crashing in some very corner case.
- A very High Priority with a Low Severity defect :
  - Slight change in logo color or spelling mistake in company name.

Defect Resolution

- **After receiving the defect report from the testing team, development team conduct a review meeting to fix defects. Then they send a Resolution Type to the testing team for further communication.**
- **Resolution Types :**
  - **Accept**
  - **Reject**
  - **Duplicate**
  - **Enhancement**
  - **Need more information**
  - **Not Reproducible**
  - **Fixed**
  - **As Designed**

Defect Triage

- **Defect triage is a process that tries to do the rebalancing of the process where test team faces the problem of limited availability of resources.**
- **When there are large number of the defects and limited testers to verify them, defect triage helps trying to get as many defects resolved based on defect parameters like severity and priority.**
- **Defect Triage Process :**



Defect Triage

- Reviewing all the defects including rejected defects by the team
- Initial assessment of the defects is based on its content and respective priority and severity settings
- Prioritizing the defect based on the inputs
- Assign the defect to correct release by product manager
- Re-directs the defect to the correct owner/team for further action

Guidelines should consider before selecting severity

- Understand the concept of priority and severity well
- Always assign the severity level based on the issue type as this will affect its priority
- Understand how a particular scenario or test case would affect the end-user
- Need to consider how much time it would take to fix the defect based on its complexity and time to verify defect.

Exercise

- **Assign the Severity for the following issues.**

1	The website performance is too slow	
2	The login function of the website does not work properly	
3	The GUI of the website does not display correctly on mobile devices	
4	The website could not remember the user login session	
5	Some links does not work.	

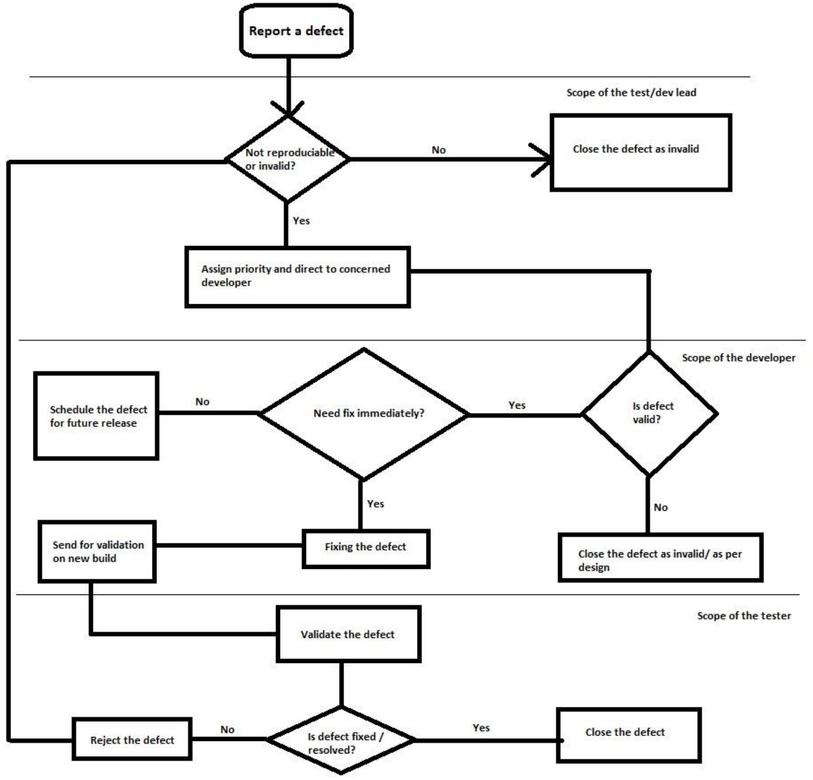
Here are the answers

No.	Description	Severity	Explanation
1	The website performance is too slow	High	The performance bug can cause huge inconvenience to user.
2	The login function of the website does not work properly	Critical	Login is one of the main function of the banking website if this feature does not work, it is serious bugs
3	The GUI of the website does not display correctly on mobile devices	Medium	The defect affects the user who use smartphone to view the website.
4	The website could not remember the user login session	High	This is a serious issue since the user will be able to login but not be able to perform any further transactions
5	Some links does not work.	Low	This is an easy fix for development guys and the user can still access the site without these links.

Tips for good Bug Report

- Structure : test carefully (Use deliberate, careful approach to testing)
- Reproduce : Test it again (rule of thumb 3 times)
- Isolate : Test it differently (Change variables that may alter symptom)
- Generalize : Test it elsewhere does the same failure occur in the other modules or locations?
- Compare : Review results of similar tests (Same test run against earlier versions)
- Summarize : Relate test to customers (Put a short tag line on each report)
- Condense : Trim unnecessary information (Eliminate extraneous words or steps)
- Disambiguate : Use Clear words (Goal : Clear, indisputable statements of fact)
- Neutralize : Express problem impartially (Deliver bad news gently)
- Review : be Sure (Peer Review)

Bug Life Cycle


Report a defect
Scope of the test/dev lead
Not reproduciable or invalid?
No
Close the defect as invalid
Yes
Assign priority and direct to concerned developer
Scope of the developer
Schedule the defect for future release
No
Need fix immediately?
Yes
Is defect valid?
Yes
No
Send for validation on new build
Fixing the defect
Close the defect as invalid/ as per design
Scope of the tester
Validate the defect
Reject the defect
No
Is defect fixed / resolved?
Yes
Close the defect

States of Defects

- New : A bug is reported and is yet to be assigned to developer
- Open : The test lead approves the bug
- Assigned : Assigned to a developer and a fix is in progress.
- Need more info : When the developer needs information to reproduce the bug or to fix the bug
- Fixed : Bug is fixed and waiting for validation
- Closed : The bug is fixed, validated and closed
- Rejected : Bug is not genuine
- Deferred : Fix will be placed in future builds
- Duplicate : Two bugs mention the same concept
- Invalid : Not a valid bug
- Reopened : Bug still exists even after the bug is fixed by the developer.

Test Cycle Closure

- **Activities :**
  - **Evaluate cycle completion criteria based on Time, Test Coverage, Cost, Software, Critical Business Objectives, Quality**
  - **Prepare test metrics based on the above parameters.**
  - **Document the learning out of the project**
  - **Prepare Test Closure Report**
  - **Qualitative and quantitative reporting of quality of the work product to the customer.**
  - **Test result analysis to find out the defect distribution by type and severity.**
- **Deliverables**
  - **Test Closure Report**
  - **Test Metrics**