Table of Contents

[Manual Testing 3](#_Toc125731721)

[**A.** **Basic Concepts:** 3](#_Toc125731722)

[1. SDLC 5](#_Toc125731723)

[**1.A**  **Spiral model:** 6](#_Toc125731724)

[**1.B**  **Prototype model feature: - blue print of the software.** 7](#_Toc125731725)

[**1.C**  **V model or VV model (verification and validation model) feature:** 7](#_Toc125731726)

[**1.C.1** **What is Static Testing?** 8](#_Toc125731727)

[**1.C.2** **Unit Testing:** 8](#_Toc125731728)

[**1.C.3** **UAT testing:** 8](#_Toc125731729)

[**1.C.4** **Dynamic testing?** 8](#_Toc125731730)

[**1.C.5** **Verification and validation?** 8](#_Toc125731731)

[**1.C.5.1** **Review** 9](#_Toc125731732)

[**1.C.5.2** **Walkthrough** 9](#_Toc125731733)

[**1.C.5.3** **Inspection** 9](#_Toc125731734)

[**1.C.5.4** **Levels of software testing** 10](#_Toc125731735)

[**1.C.5.5** **Unit Testing** 10](#_Toc125731736)

[**1.C.5.6** **Unit testing techniques** 10](#_Toc125731737)

[**1.C.5.7** **Integration testing** 11](#_Toc125731738)

[**1.C.5.8** **Types of Integration Testing** 11](#_Toc125731739)

[**1.C.6** **System Testing** 11](#_Toc125731740)

[**1.C.7** **UAT Testing** 12](#_Toc125731741)

[**1.A.1** **System Testing** 13](#_Toc125731742)

[**1.A.1.1** **Graphical user-Interface Testing or GUI Testing:** 13](#_Toc125731743)

[**1.A.1.2** **Usability Testing:** 14](#_Toc125731744)

[**1.A.1.3** **Functional Testing** 14](#_Toc125731745)

[**1.A.1.4** **Non-functional Testing:** 16](#_Toc125731746)

[**1.A.2** **Regression Testing:** 18](#_Toc125731747)

[**1.A.3** **Re Testing:** 18](#_Toc125731748)

[**1.A.4** **Exploratory Testing:** 20](#_Toc125731749)

[**1.A.5** **Adhoc Testing:** 20](#_Toc125731750)

[**1.A.5** **Monkey/Gorilla Testing:** 21](#_Toc125731751)

[**1.A.6** **Positive Testing:** 21](#_Toc125731752)

[**1.A.7** **Negative Testing:** 22](#_Toc125731753)

[**1.A.8** **End – to – End Testing:** 22](#_Toc125731754)

[**1.A.9** **Globalization and Localization Testing:** 22](#_Toc125731755)

[2. STLC 28](#_Toc125731756)

[**2.1** **Test plan Contents:** 30](#_Toc125731757)

[**2.2** **Use case, Test Scenario & Test Case** 30](#_Toc125731758)

[**2.3** **Test Suit:** 32](#_Toc125731759)

[**2.3.1** **Requirement Traceability Matrix (RTM)** 34](#_Toc125731760)

[**2.3.2** **Test Environment:** 35](#_Toc125731761)

[**2.3.3** **Test Execution:** 35](#_Toc125731762)

[**2.3.4** **Guideline for Test Execution** 35](#_Toc125731763)

[**2.3.5** **Defect Severity** 37](#_Toc125731764)

[**2.3.6** **Defect Priority** 37](#_Toc125731765)

[**2.3.7** **Defect Resolution:** 39](#_Toc125731766)

[**2.4** **Bug Life Cycle:** 40](#_Toc125731767)

[**2.5** **Test Cycle Closure:** 40](#_Toc125731768)

[**2.5.1** **Test Metrics** 41](#_Toc125731769)

[**2.5.2** **QA / Testing Activities** 43](#_Toc125731770)

[**2.5.3** **7 Principles of Software Testing:** 43](#_Toc125731771)

# **Manual Testing**

## **Basic Concepts:**

1). **What is software?**

Ans. A Software is a collection of compute programs that helps us to perform a task.

2). **What is program?**

Ans. A program is a coding, these are some instructions that is understandable by the computer.

3). **How many types of software?**

Ans. There are 3 types of software used in the computer.

a). System Software:

Example: Device drivers, Operating Systems, Servers, Utilities etc.

b). Programming software:

Example: Compilers, debuggers, Interpreters etc.

c). Application software:

Example: Web Application, Mobile Apps, Desktop Applications etc.

4). **What is software testing? And why it is necessary?**

Ans. 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.

5). **What is software Quality?**

Ans. A software is a quality product based on some parameters; these are:

Bug-free.

Delivered on time.

Within Budget.

Meets requirements and / or expectations.

Maintainable.

6). **What is project and products?**

Ans. If software application is developed for **specific customer** based on the requirement then it is called project.

If software application is developed for **multiple customers** based on market requirements then it is called product.

Example: If we develop a software for a specific Bank for their specific requirements, this software will use only this Bank for their work. It is called project.

WhatsApp and MS word these are the example of product. Everyone can use this software.

**There are two types of companies these are:**

Service based company and product-based company.

Service based companies, most of the time they deal with the projects, they will work with the specific customers/clients. Example: TCS

Google is a product-based company, Microsoft, Oracle all are product-based company. They all have multiple application in the market.

7). **Why do we need testing?**

Ans. We have to deliver the quality product to the customer and ensure that our software has no bug and fulfil the customer requirements.

8). **Error, bug and failure?**

Ans. Error is something nothing but a human mistake, incorrect human action.

Example: human mistake, typo in writing code is an example.

Bug: The deviation of actual behavior and expected behavior is called bug / Defect.

Failure: After releasing the software in the customer environment. The deviation identified by the end user while work the software is called failure.

9). **Why the software has bug?**

Ans. There are lot of reasons we have, but mainly 5 reasons we can say:

1. Miscommunication or no communication
2. Software complexity
3. Programming errors
4. Changing requirements
5. Lack of skilled testers

**Video # 2**

# **SDLC**

1). **What is SDLC?**

Ans. It is Software Development Life Cycle. Which is a process uses by software industry to Design, Develop and Test the software.

For any company, 3 p’s are very important:

P- People – every company should some people

P-Process – there should be some process

P-Product- Deliver product to customer

There are some phases of SDLC by which it works

1. Requirement analysis
2. Design
3. Development
4. Testing
5. Maintenance

Every company have followed some models, one of the models is Waterfall Model:

It is very old and conventional model. Now a days no company is following this model.

But for interview prospect we have to know this.

It will start with the Requirement analysis, activity will be documented in every step, documentation will play a very important role. Business people in this stage communicate between customer and the company. They will collect the requirements and after then they will prepare some documents, which is called **SRS (Software Requirement Specification)** document.

Once the requirement got then the system design is started, they will divide the entire requirement in High level modules and low-level modules.

One the design phase is completed then the implementation phase will start based on the design document. Coding is started in this phase.

After the implementation, testing phase will start. In this phase tester will test the software.

After successfully complete the testing, they install the software in the customer environment and start using the software, this is deployment.

When start using the software, maintenance is started.

In this waterfall model, every phase depends on the previous phase result. Without completion of previous phase next step cannot be started.

**Advantage of Waterfall model:**

1. The quality of the product will be very good. Because all are well documentation.
2. Since the Requirement changes are not allowed, chances of finding bugs will be less.
3. Initial investment is less since the testers are hired at the later stages.
4. Preferred for small projects where requirements are freeze.

**Disadvantages of Waterfall model:**

1. Requirement changes are not allowed
2. If there is defect in requirement that will be continued in later phases.
3. Total investment is more because time taking for rework on defect is time consuming which leads to high investment.
4. Testing will start only after coding.

### **1.A Spiral model:**

----------------------

To overcome the disadvantages of Waterfall model, Spiral model introduced.

In this model, (Planning + Risk Analysis) is the Requirement Analysis, Engineering and Execution is the Design and Development and Evaluation is the Testing same as Waterfall model.

Here every cycle completed all complete cycle and send the software to customer to use that is called version 1. After then if the customer changes the requirements then again complete the full cycle and send the software to customer which is second version. Like this, every complete cycle it will deliver a new version.

WhatsApp, Windows are the example of spiral model. Every time it comes to market with new features.

**Spiral model feature:**

------------------------------------------------------

1. Spiral model is an iterative model.
2. Spiral model overcome drawbacks of Waterfall model.
3. We follow spiral model whenever there is dependency on the modules.
4. In every cycle new software will be released to customer.
5. Software will be released in multiple versions. So it is also called version control model.

**Advantages of Spiral model:**

------------------------------------------------------

1. Testing is done in every cycle, before going to next cycle.
2. Customer will get to use the software for every module.
3. Requirement changes are allowed after every cycle before going to the next cycle.

**Disadvantages of Spiral model:**

------------------------------------------------------

1. Requirement changes are not allowed in between the cycle.
2. Every cycle of spiral model looks like waterfall model.
3. There is no testing in requirement & design phase.

### **1.B Prototype model feature: - blue print of the software.**

------------------------------------------------------

Firstly, requirements received from the customer then make a protype (sample copy) software.

If the customer accepts this model then started the designing, coding and testing.

It is in between Waterfall and Spiral model.

Any software is divided into different modules, the entire software is not a single piece, it is combined of modules. Different teams can work in different modules.

V-model is a very important model

### **1.C V model or VV model (verification and validation model) feature:**

------------------------------------------------------

In every phase we do the testing, example: in requirement analysis phase we do the test, in development phase we do the test, in testing phase we do the test. This is the specialty of this model.

There is some terminology used here:

These documents will be prepared by the business people, these are understandable by the business people, not by the technical people. These are the requirements from the customer or stakeholder. Business unit will make this document.

BRS: Business Requirement Specifications

CRS: Customer Requirement Specifications

URS: Users requirement Specifications.

These 3 documents can be converted to SRS (Software Requirements Specifications) this document is understandable by the technical people. SRS is created by project manager / product manager.

Based on this SRS designer create HLD (High level module) and LLD (Low level module) by designers. HLD contains the main modules and LLD contains the low level modules.

In the first phase, prepared documents by the business people, they are the responsible for testing the documents. This is not test the software, they will test that all the documents are correct or not, this is the testing. If the documents are correct then there are will be no difficulties in the next steps. It is review. We are testing the documents in the form of review.

By using review, walkthrough or inspection, by using these three techniques we have tested. This is called Static Testing.

### **1.C.1 What is Static Testing?**

Ans. Testing the project related documents is called as Static testing. In the form of review, walkthroughs and inspections.

**1.C.2 Unit Testing:** Testing of a single module is called unit testing. Developers will conduct this testing.

After combining all module then the complete software build, the testing of this complete software is called **Integration Testing.** This is also conducted by the developers.

If anyone want to conduct these unit testing and integration testing, they have to know the coding / programming knowledge. This type of testing is called **white box** testing.

In the system testing phase, the testers are responsible for this test. It is called **Black Box** testing techniques.

**1.C.3 UAT testing:** This is done by tester along with the customer.

### **1.C.4 Dynamic testing?**

Testing the actual software. We are using unit testing, integration testing, System testing, User acceptance testing.

### **1.C.5 Verification and validation?**

Verification checks whether we are building the right product.

Focus on documentation.

Verification typically involves, reviews, Walkthroughs or inspections.

Static testing techniques uses here.

Validation checks whether we are building the product right.

Takes place after verifications are completed.

Focus on Software

Validation typically involves actual testing.

Unit testing, integration, system testing, UAT testing

**Advantages of V model:**

------------------------------------------------------

1. Testing is involved in each and every phase.

**Advantages of V model:**

------------------------------------------------------

1. Documentation is more
2. Initial investment is more.

**Static testing techniques:**

------------------------------------------------------

1. Review
2. Walkthrough
3. Inspection

**Dynamic testing techniques:**

------------------------------------------------------

1. Unit testing
2. Integration testing
3. System testing
4. UAT (User Acceptance Test).

### **1.C.5.1 Review**

----------------

It is conducts on the documents to ensure correctness and completeness.

There are various kinds of review we have to do:

1. Requirements review
2. Design reviews
3. Code reviews
4. Test plan reviews
5. Test cases reviews etc.

Anybody can do the review any time, he can be manager, developer or tester.

### **1.C.5.2 Walkthrough**

----------------

1. It is an informal review.(Informal means, there is no schedule for that meeting
2. Author (who create the document) reads the documents or code and discuss with peers.
3. It’s not pre-planned and can be done whenever required.
4. Also, walkthrough does not have minutes of the meet.

### **1.C.5.3 Inspection**

----------------

1. Its most formal review type.
2. In which al least 3-8 people will sit in the meeting, 1-reader, 2-writer, 3-moderator plus concerned.
3. Inspection will have a proper schedule which will be intimated via email to the concerned developer/tester.

**Difference between QA and QC**

--------------------------------------------

QA is process related

QC is the actual testing of the software

QA focuses on building in quality.

QC focuses on testing for quality.

QA is preventing defects.

QC is detecting defects.

QA is process oriented.

QC is product oriented.

QA for entire life cycle.

QC for testing part in SDLC.

QE 🡪 is Quality Engineering. They will write the code for testing the software, whereas software engineers, write the code for building/ development the software.

### **1.C.5.4 Levels of software testing**

-----------------------------------------

Unit testing

Integration testing

System testing (overall functionality testing of the application)

User acceptance testing (UAT).

### **1.C.5.5 Unit Testing**

----------------------

A unit is a single component or module of a software.

Unit testing conducts on a single program or single module.

Unit testing is white box testing technique.

Unit testing is conducted by the developers.

### **1.C.5.6 Unit testing techniques**

-----------------------------------

Basic path testing

Control structure testing

Conditional coverage

Loops coverage

Mutation testing.

### **1.C.5.7 Integration testing**

-----------------------------------

1. Integration testing performed 2 or more modules.
2. Integration testing focuses on checking data communication between multiple modules.
3. Integrated testing is white box testing technique.

The developer and tester both can conduct integration testing, the developer can test their code inside the program but the tester can test this in the application level where the UI already built.

### **1.C.5.8 Types of Integration Testing**

---------------------------------------

1. **Incremental Integration testing** (Incrementally adding the modules and testing the data flow between the modules). There are 2 Approaches in this Incremental Integration testing, (a). Top Down and (b). Bottom up.
2. Top-Down Approach: Incrementally adding the modules and testing the data flow between the modules and Ensure the module added is the child of previous module.
3. Bottom-Up Approach: Incrementally adding the modules and testing the data flow between the modules and Ensure the module added is the parent of previous module.
4. Sandwich/Hybrid Approach: Combination of Top-Down & Bottom up Approach is called Sandwich Approach.
5. **Non-Incremental Integration testing**. Adding all the modules in a single shot and test the data flow between modules.

Non-Incremental Integration testing is not preferable by the company due to some drawbacks.

**Drawbacks are:**

1. We might miss data flow between some of the modules.
2. If we find any defects, we cannot understand the root cause of the defect.

**1.C.6 System Testing. (Here the testers are engaged)**

--------------------------

1. Testing over all functionality of the application with respective client requirements.
2. It is a Black-Box testing technique.
3. This testing is conducted by testing team.
4. After completion of component and integration level testing’s we start system testing.
5. Before conducting system testing, we should know the customer requirements.
6. System Testing focuses on below aspects.
7. User Interface Testing (GUI).
8. Functional Testing.
9. Non-Functional Testing.
10. Usability testing.

### **1.C.7 UAT Testing**

--------------------------

After completion of system testing UAT team conducts acceptance testing in two levels.

**Alpha testing** (When the customer comes to the development environment and test the software, then it is called Alpha Testing)

**Beta testing** (After finishing the Alpha testing when the customer install the software in customer environment and test that, it is called Beta testing).

**Video no. 4**

## **1.A.1 System Testing**

--------------------------

System testing is nothing but an overall functionality test of the software with respect to customer requirement.

In system testing we are usually testing in 4 different areas

1. GUI Testing
2. Usability Testing
3. Functionality Testing
4. Non-Functional Testing. (In this testing we need separate skill and environment, so, the team will be different).

**1.A.1.1 Graphical user-Interface Testing or GUI Testing:** It is a process of testing the user interface of an application.

**A Graphical user interface** includes all the elements such as menus, Checkbox, buttons, colors, fonts, sizes icons, contents and images.

**GUI Testing checklist:**

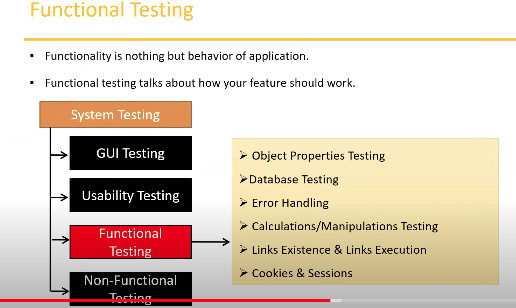
* Testing the size, position, width, height of the elements.
* Testing of the error messages that are getting displayed.
* Testing the different sections of the screen.
* Testing the font whether it is readable or not.
* Testing of the screen in different resolutions with the help of zooming in and zooming out.
* Testing the alignment of the texts and other elements like icons, buttons, etc. are in proper place or not.
* Testing the colors of the fonts.
* Testing whether the image has good clarity or not.
* Testing the alignment of the images.
* Testing of the spelling.
* The user must not get frustrated while using the system interface.
* Testing whether the interface is attractive or not.
* Testing of the scrollbars according to the size of the page if any.
* Testing of the disabled fields if any.
* Testing of the size of the images.
* Testing of the headings whether it is properly aligned or not.
* Testing of the color of the hyperlink.
* Testing UI Elements like button, textbox, text area, check box, radio buttons, drop downs, links etc.

### **1.A.1.2 Usability Testing:**

--------------------------

a). During this testing validates application provided context sensitive help or not to the user.

b). Checks how easily the end users are able to understand and operate the application is called usability testing.

****

### **1.A.1.3 Functional Testing**

--------------------------------

Functionality is nothing but behavior of application.

Functional testing talks about how your feature should work.

1. **Object Properties Testing** : Check the properties of objects present on the Application. Ex. Enable, disable, visible, Focus etc.
2. **Database Testing / Backend Testing**: It is DML (Data Manipulation Language) Operation. For this operation we need the knowledge of SQL, because retrieving, deleting, inserting data in the backend of the software is tested here. In every software in the front-end UI is there and, in the backend, data base is there, so, how this data base is working or not we have to check. This checking or testing is done by Data Manipulation Language Testing. Here from the front end (Black Box) we are sending command to the backend or data base (White Box), so, the data base testing is **Grey Box testing**, i. e. combination of black box and white box testing.

Insert, update, delete and select is the basic knowledge of SQL Other than we should know

Table and column level validation (Column type, column length, number of columns ……), relation between the tables (normalization), functions, Procedures, Triggers, Indexes, views etc.

1. **Error Handling:** Verifying the error messages while performing incorrect actions on the application. Error messages should be readable and easy understandable to the user.
2. **Calculations/Manipulations Testing**: Tester should verify the calculations.
3. **Links Existence & Links execution**: These two tests (Links existence and Links execution and Cookies & Session) both are used for web application.

**Links existence means**: Where exactly the links are placed and

**Links execution means**: Links are navigating to proper page or not.

There are three types of links:

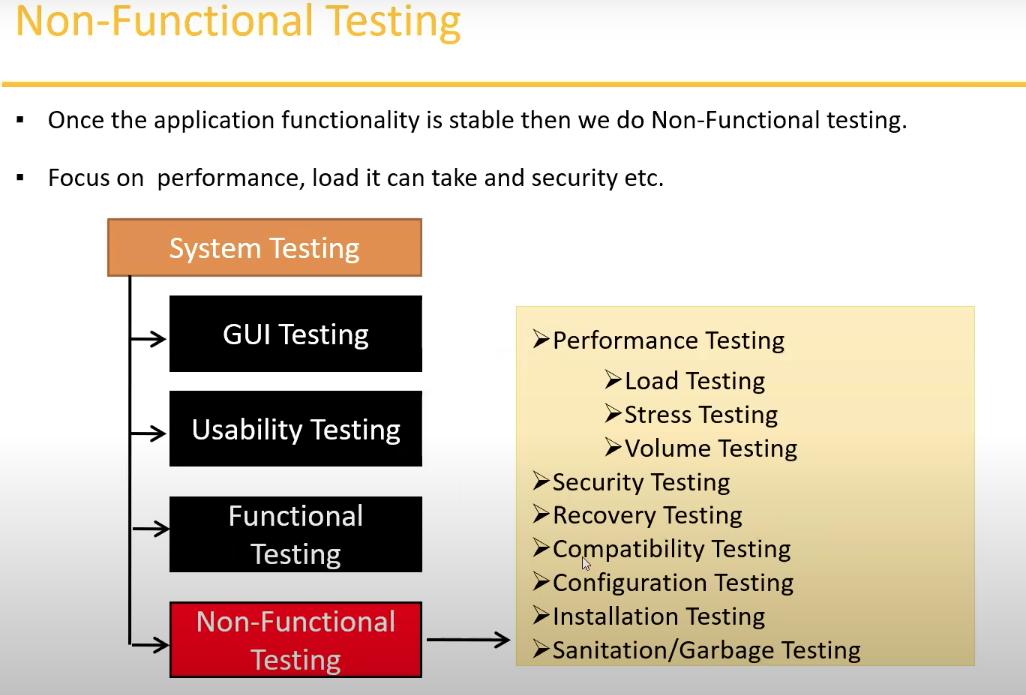
1. **Internal links**, it means when clicks on the link, it will navigate to the same page, but different section.
2. **External links**, means when clicks it will navigate to some other page.
3. **Broken links** means when click on the links, it will not navigate to any other pages or any where else. But the link is there. It does not have target page. For future use it is used.
4. **Cookies & Sessions**:

**Cookies**: Cookies are nothing but some temporary files which are saved by the browser, while we browsing the pages through internet. Whenever we are going to test the cookies, we have to test that the browsers are creating the cookies, saving data or not. It is used for only for web application. It is in client site.

**Sessions**: Sessions will be created on the server side. When we worked on a page and stopped working on that page, there is one timer start counting time, if there is a fixed time to stopped the page then it will again ask for the credential again. This **time frame** is called session. This is used for security reason. It will show session is expired. It is in the server site.

What is **Focus attribute**?

Ans. When insert the username, after then it automatically comes to the password field and blinks the cursor there. This action is called Focus attribute.



**1.A.1.4 Non-functional Testing:** Ones the functional testing is stable, then starts the non-functional testing. It is tested the customer’s expectation. Performance, load, security etc. For this test it will need special people. Special environment also needed.

**Performance** : It means the speed of the application. Performance testing is always do for web application. For desktop application no need performance testing. In the performance testing we are testing these tests:

1. **Load**: in this testing we will increase the number of user up to a certain number, every time we have to calculate the speed of the application, that means the response time from the server. Increasing the load on the application slowly and then check the speed of the application. There are some tools, by that we can increase the number of users. Jmeter is the example of that tools. At what number the application breaks, that is the bench mark.
2. **Stress:** In load testing, gradually increase the load but in Stress testing we suddenly increase/decrease the load on the application. To check the application is break or not. The sequence like, 5🡪20🡪100 🡪50 like this suddenly increase or decrease the load.
3. **Volume**: It is nothing but size. How much data the application can handle.

**Security testing:** How secure the application. It is done by the specialist. Under this testing we have to test:

1. **Authentication:** The valid users can access or not. Test that the user is valid or not.
2. **Authorization / Access Control**: The valid users can access the application, he can access only some features. This is controlled by the authority. Permission of the valid users.

**Recovery Testing:** Whether the application is recovering or not. If by chance the users delete something, can it recover or not. Check the system change to abnormal to normal.

**Compatibility Testing:**

1. **Forward compatibility**: If we have installed one version of application in the machine, then next version should compatible to the previous version. The old software should upgrade to new software.
2. **Backward compatibility**: Opposite of Forward compatibility.
3. **Hardware compatibility / Configuration Testing**: As per the requirements, the application can run on that machine or not. It supports the particular RAM, OS or other hardware etc.
4. **Installation Testing**: Check the installation process, each and every screen is coming properly or not. Screens are understandable or not. The uninstallation process is clear or not. Is there any file remain in the system after the uninstallation? Everything should in the installation process.
5. **Sanitation or Garbage Testing**: If any application provides extra features / functionality then we consider as bugs. We have to remove the unwanted stuff. This type of testing is called Sanitation or Garbage testing.

Q: If there any extra functionality available in the application, should we consider that as a bug?

Ans. Yes, it is considered as a bug. We should report that and remove that functionality. It is called Sanitation or Garbage testing.

Q. What is the difference between functional and non-functional testing?

Ans.

|  |  |
| --- | --- |
| **Functional Testing** | **Non-Functional Testing** |
| Validates functionality of software | Verify the performance, security, reliability of the software. |
| Functionality describes what software does. | Non-functionality describe how software works. |
| Concentrates on user requirement. | Concentrate on users expectations. |
| Functional testing takes place before Non-functional testing. | Non-functional testing performed after finishing Functional testing. |

**Video # 5**

## **1.A.2 Regression Testing:**

----------------------------

Testing conducts on modified build to make sure there will not be impact on existing functionality because of changes like adding/deleting/modifying features.

**Unit regression testing:**

Testing only the changes/modifications done by the developer.

**Regional Regression Testing**:

Testing the modified module along with the impacted modules

**Impact Analysis meeting** conducts to identify impacted modules with QA & Dev.

**Full Regression:**

Testing the main feature & remaining part of the application.

Example: Dev has done changes in many modules, instead of identifying impacted modules, we perform one round of full regression. Almost 80% of software changes when the Dev fixes the bug. At that time Impact Analysis meeting will be wasting of time. For that case we will test all of the modules and that time it is called Full Regression testing.

## **1.A.3 Re Testing:**

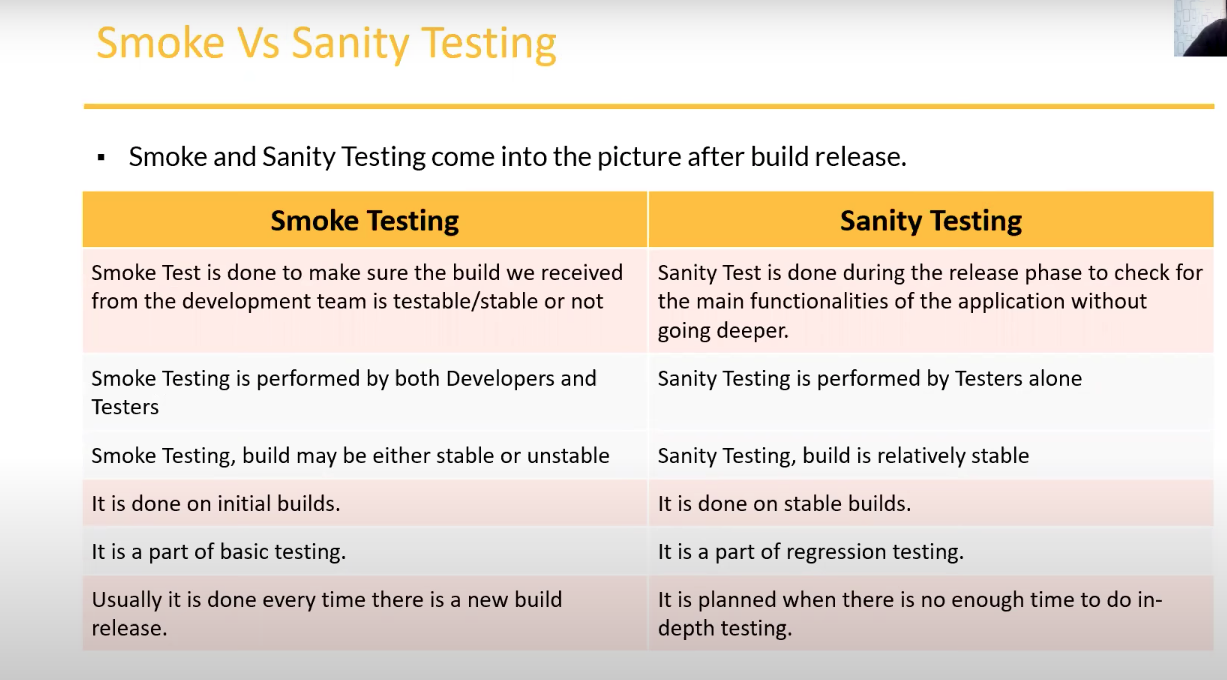
----------------------------

* Whenever the developer fixed a bug, tester will test the bug fix is called Re-testing.
* Tester close the bug if it worked otherwise re-open and send to developer.
* To ensure that the defects which were found and posted in the earlier build were fixed or not in the current build.
* Example:
  + 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.

Example : Re-Testing Vs Regression Testing:

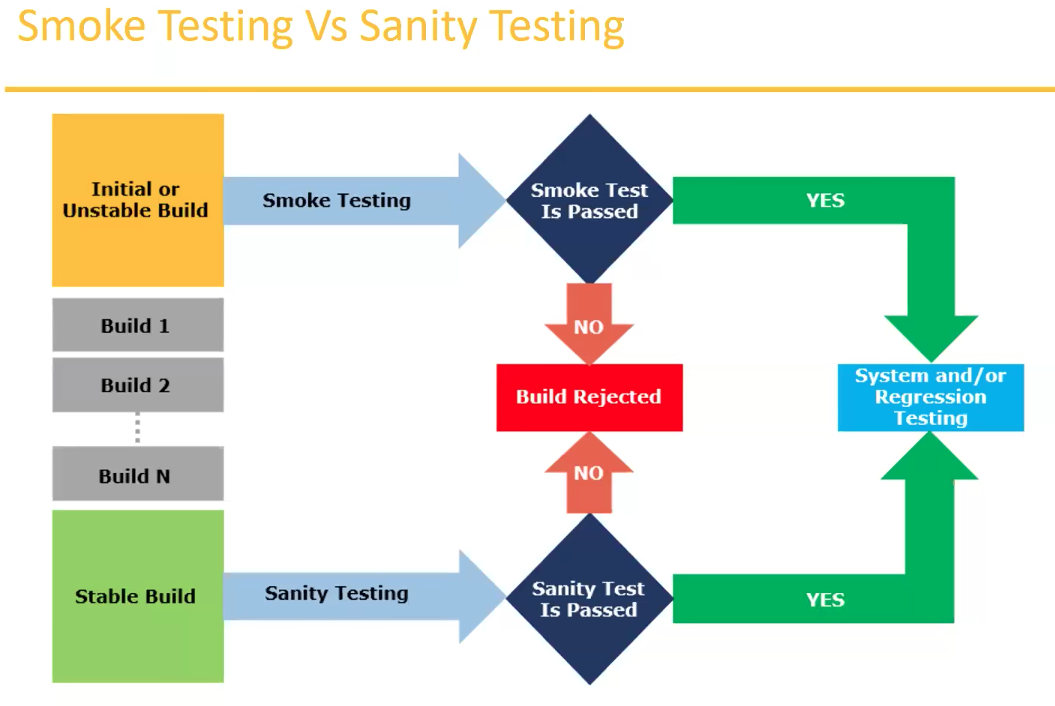
* An Application under test has three modules namely Admin, Purchase and Finance.
* Finance module depends on Purchase module.
* If a tester found a bug on Purchase module and posted. Once the bug is fixed, the tester needs to do **Retesting** to verify whether the bug related to the Purchase is fixed or not and also tester needs to do **Regression Testing** to test the Finance module which depends on the Purchase module.

**Smoke Vs Sanity Testing:**



In Smoke testing we will conduct the basic feature on the application, such as installation is doing good or not and navigate the pages, this type of testing is called Smoke testing. There is no test cases written for this test. In the very begging of the build every time the tester tests that installation is doing correctly or not. Pages are navigate or not. Initial build very frequently they test these. This is called Smoke testing. In this stage the build is unstable. Smoke testing is done by the developer and testers also.

After the smoke test, when the build is stable after the unstable stage. The basic navigation and basic functionality testing like, the links are working, the name of the product is shown, log in and logout is working etc. Not in deep testing only the basic functionality. Sanity testing is done by the testers only.



## **1.A.4 Exploratory Testing:**

* We have to explore the application, understand completely and test it.
* Understand the application, identify all possible scenarios, document it then use it for testing.
* We do exploratory testing when the Application ready but there is no requirement.
* Test Engineer will do exploratory testing when there is no requirement.

**Drawbacks:**

* You must misunderstand any feature as a bug (or) any bug as a feature since you do not have requirement.
* Time consuming
* If there is any bug in application, you will never know about it.

## **1.A.5 Adhoc Testing:**

Depends on tester’s previous experience, an application is given, then he tests the application without any document or test cases. It is a randomly test the functionality. Intention to break the application to find out the Corner scenarios.

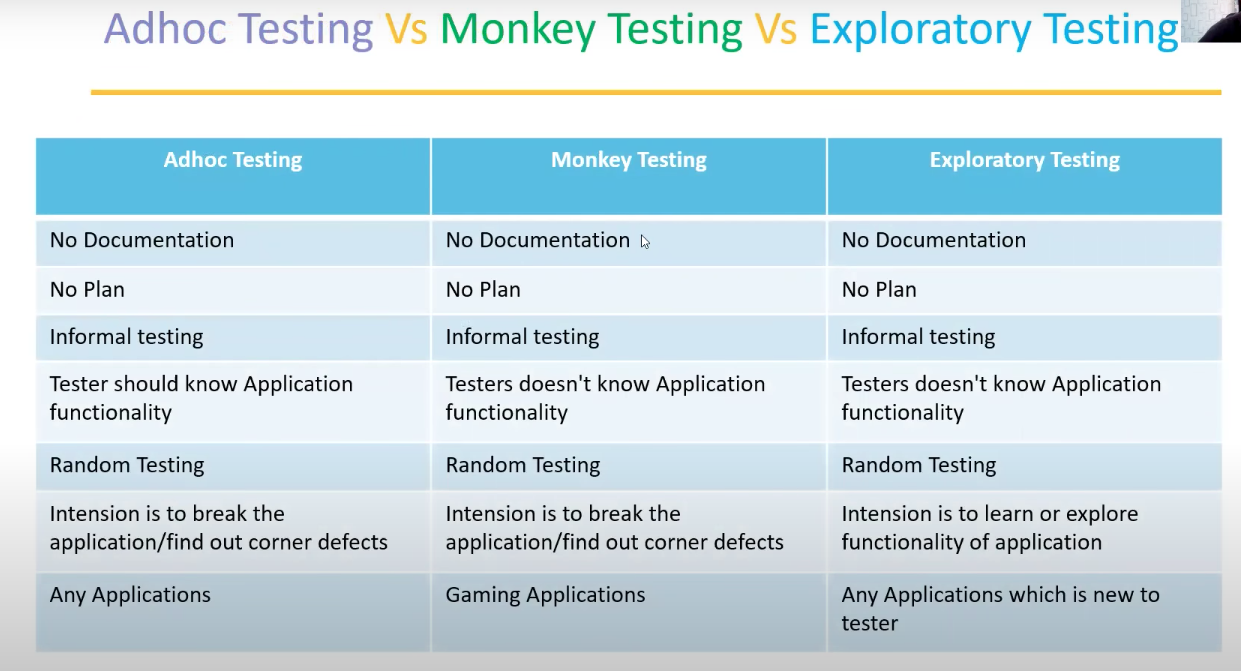
* Testing application randomly without any test cases or any business requirement document.
* Adhoc testing is an informal testing type with an aim to break the system.
* Tester should have knowledge of application even though he doesn’t have requirements/test cases.
* This testing is usually an unplanned activity.

## **1.A.5 Monkey/Gorilla Testing:**

The tester have no experience about this application. Such as a new gaming application anybody can click any link or input anything or enter any data, at that time it will not break the application.

* Testing application randomly without any test cases or any business reequipment document.
* Adhoc testing is an informal testing type with an aim to break the system.
* Tester do not have knowledge of application.
* Suitable for gaming application.

**Difference between Adhoc Testing, Monkey Testing and Exploratory Testing:**



## **1.A.6 Positive Testing:**

-----------------------------

* Testing the application with valid inputs is called as Positive Testing.
* It checks whether an application behaves as expected with positive inputs.

Example:

There is a text box in an application which can accept only numbers. Entering values up to 99999 will be acceptable by the system and any other values apart from this should not be acceptable.

To do positive testing, set the valid input values from 0 to 99999 and check whether the system is accepting the values.

## **1.A.7 Negative Testing:**

--------------------------

* Testing the application with invalid input is called as negative Testing.
* It checks whether an application behaves as expected with the negative inputs.

Example:

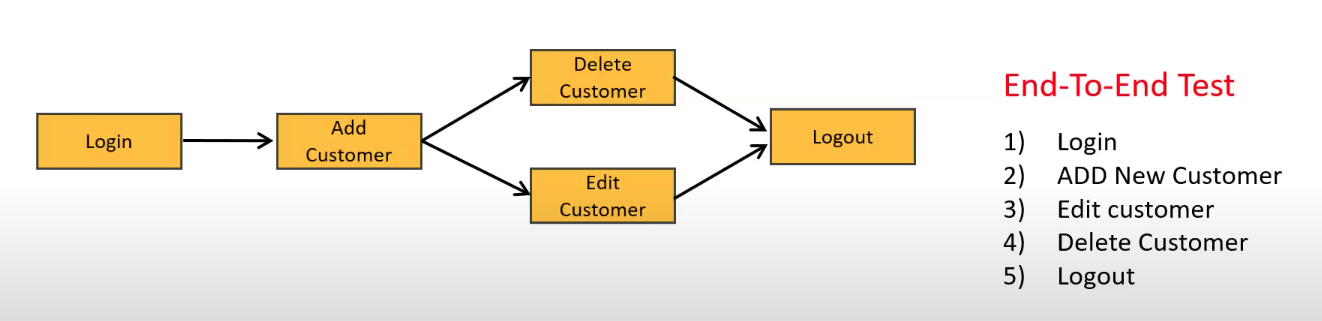
Negative testing can be performed by entering characters A to Z from a to z.

Either software system should not accept the values or else it should throw an error message for these invalid data inputs.

## **1.A.8 End – to – End Testing:**

------------------------------------

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



## **1.A.9 Globalization and Localization Testing:**

---------------------------------------------------------------

**Globalization testing it is also called Internationalization (i18n)Testing:**

* Performed to ensure the system or software application can run in any cultural or local environment.
* Different aspects of the software application are tested to ensure that it supports every language and different attributes.
* It tests the different currency formats, mobile number formats and address formats are supported by the application.
* **For example,** Facebook.com supports many of the languages and it can be accessed by people of different countries. Hence it is a globalized product

**Localization Testing:**

-------------------------------

* Performed to check system or software application for a specific geographical and couturial environment.
* Localized product only supports the specific kind of language and is usable only in specific region.
* It tests the specific currency format, mobile number format and address format is working properly or not.
* **For example**, Baidu.com supports only the Chinese language and can be accessed only by people of few countries. Hence it is a localized product.

**Video # 6**

**Test design Techniques / Test Data Design Techniques / Test Case Design Techniques**

-----------------------------------------------------------------------------------------------------------------------------

Used to prepare data for testing. It is used to reduce more data and more coverage.

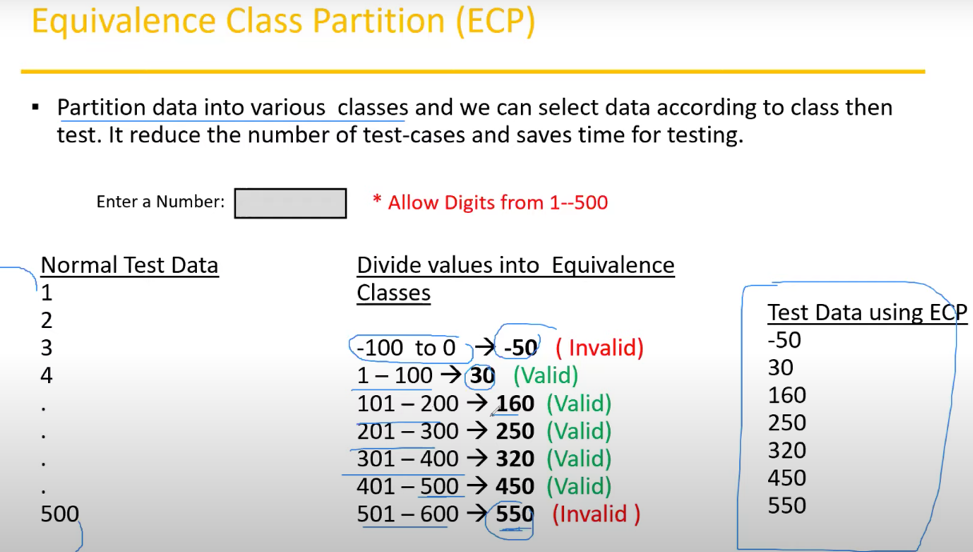
1). Reduce the Data

2). More Coverage

There are 5 Techniques we can use to achieve these.

* 1. **Equivalence Class Partitioning (ECP):**

We can simply divide the data into multiple classes it is called Equivalence class partitioning. Each partition can pick any data from that class. It will pick randomly any number from that class.



There is another example of ECP



Value Check

Classify / Divide / Partition the data in to multiple classes.

* 1. **Boundary Value Analysis (BVA):**

Here we will consider the boundary of the values. We will test only the boundaries. The technique is that First input the minimum number, it should accept, then minimum plus 1, it should accept, after then minimum minus 1, it should not accept. Similar way use for the maximum value. So, in both ends we can use only 6 values. That will cover all the boundaries.

**Input domain testing:**

The value will be verified in the text box/input fields.

We use ECP & BVA techniques it is called Input Domain Testing.

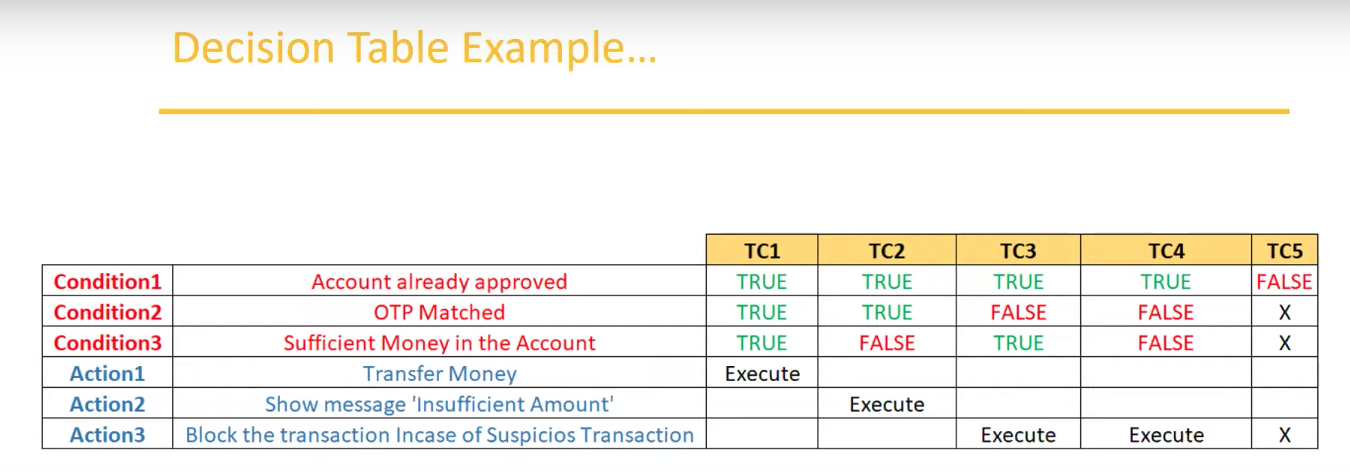
* 1. Decision Table Based testing.
* Decision table is also called as Cause-Effect Table.
* This technique will be used if we have more conditions and corresponding actions.
* In Decision table technique, we deal with combinations of inputs.
* To identify the test cases with decision table, we consider conditions and actions.

**Decision Table Example:**

If we have more number of conditions / actions then we use decision table technique.

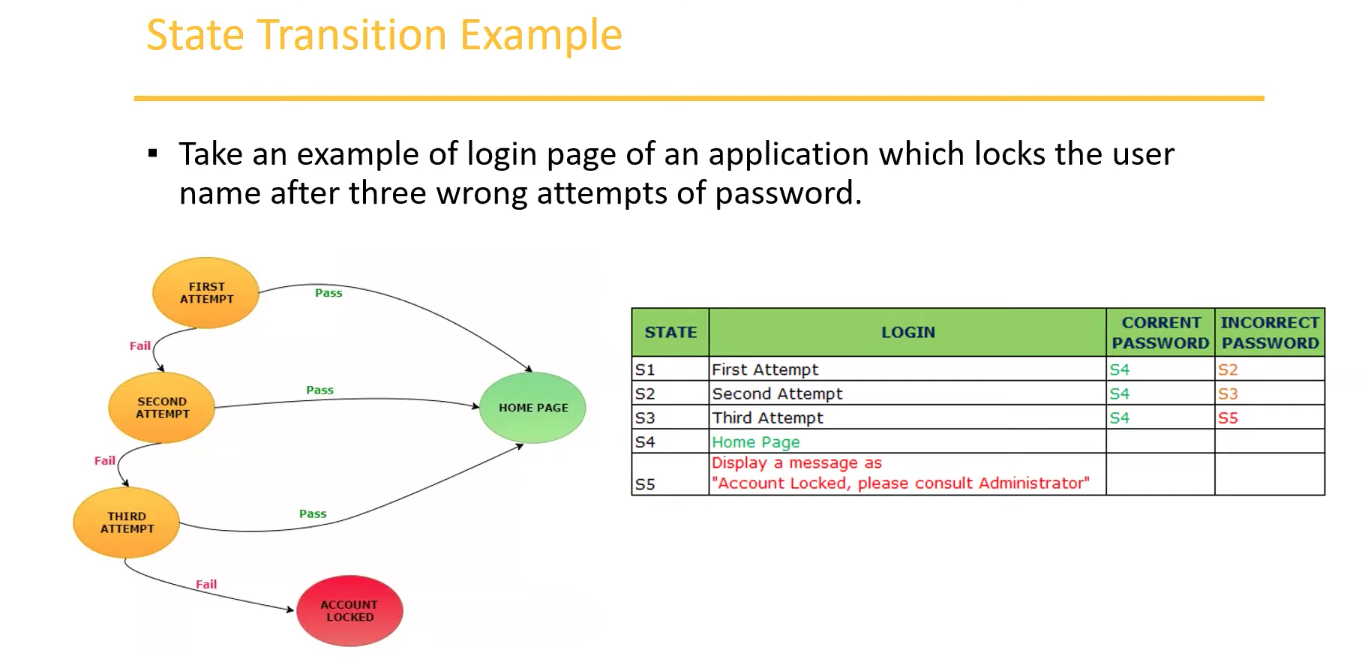
* Take an example of transferring money online to an account which is already added and approved.
* Here the conditions to transfer money are
  + Account already approved
  + OTP (one time password) matched
  + Sufficient money in the account
* And the actions performed are
  + Transfer money
  + Show a message as insufficient amount
  + Block the transaction incase of suspicious transaction

As per this example, if we fulfil the certain conditions then we can transfer money. Whenever we have these kinds of conditions then we have to use decision table technique.



* 1. **State Transition:**
* In state Transition technique changes in input conditions change the state of the Application.
* This testing technique allows the tester to test the behaviors of an AUT.
* 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 behavior.

Example of State Transition Example:



* 1. **Error Guessing:**
* Error guessing is one of the testing techniques used to find bugs in a software application based on tester’s prior experience.
* In error guessing we don’t follow any specific rules.
* It depends on Tester Analytical Skills and experience.
* Some of the examples are:

a). Submitting a form without entering values.

b). Entering invalid values such as entering alphabets in the numeric field.

**Video # 7 (Starting of STLC)**

# **STLC**

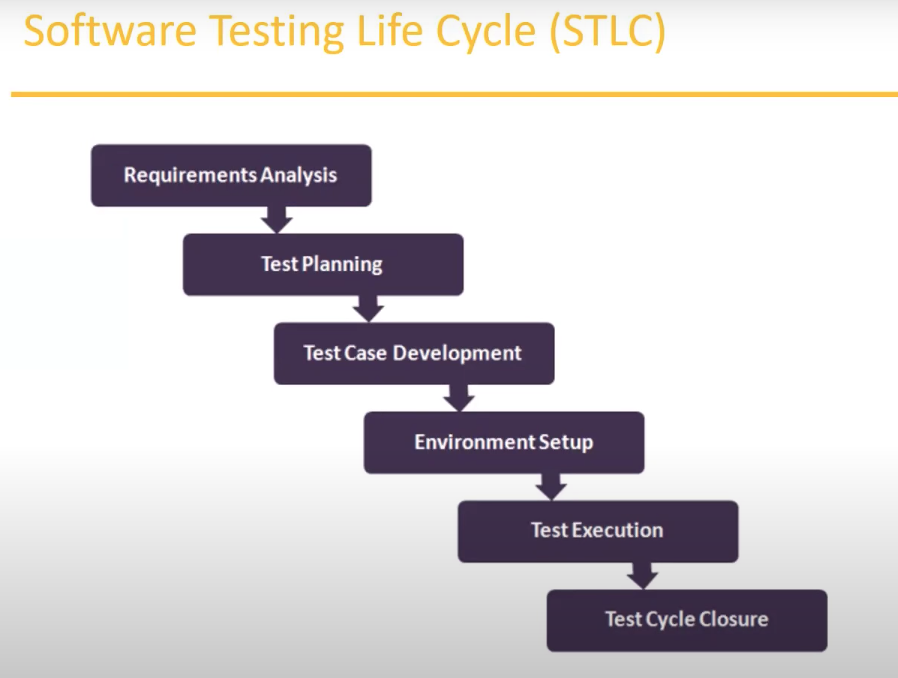
SDLC: Requirement Analysis, Design, Coding, Testing, Deployment, Maintenance

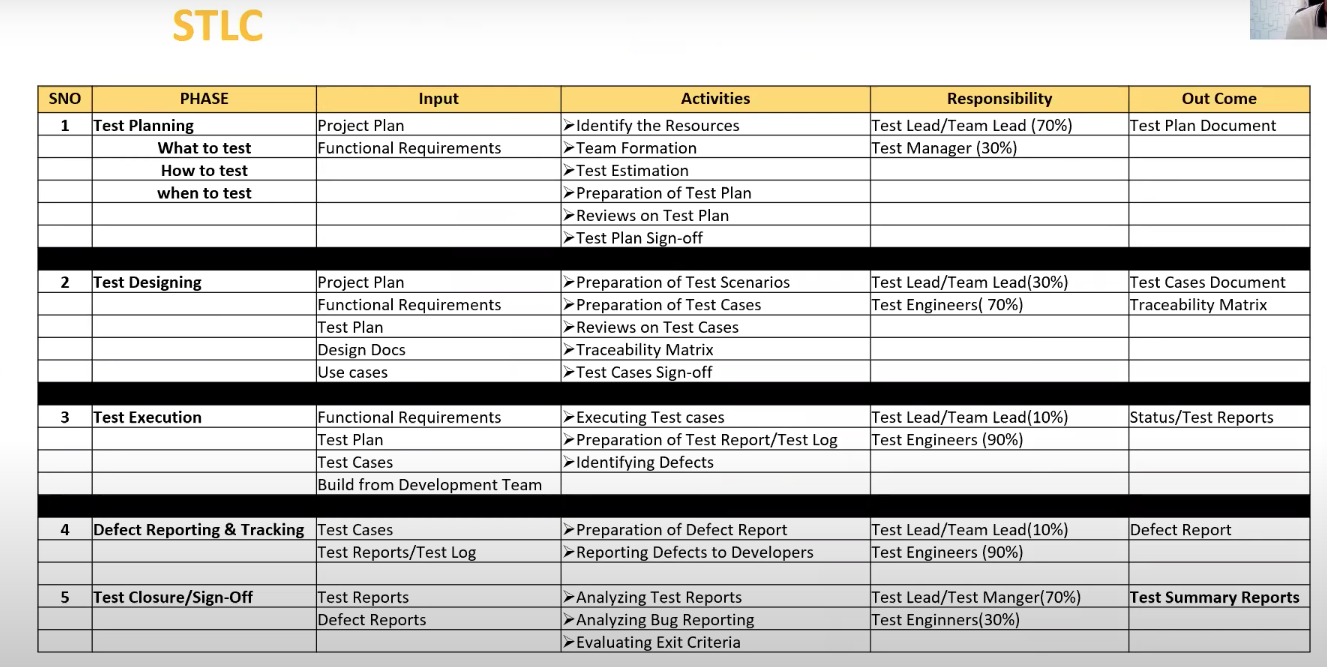
STLC is the part of SDLC, and the different phases under the STLC will conduct. STLC is the process.

STLC: Software Testing Life Cycle

Phases of STLC:

1. Requirement Analysis
2. Test Planning
3. Test Design
4. Test Execution
5. Defect /Bug Reporting & Tracking
6. Test closure





Project plan is prepared by the project manager, all the time frame, project requirements all other high-level documents will be there. FRS documents, lead level or manager level staff will prepare this.

**Video # 8**

## **2.1 Test plan Contents:**

* A Test Plan is a document that describes the test scope, test strategy, objectives, schedule, deliverables and resources required to perform testing for a software product.
* Test plan template contents:
  + Overview
  + Scope
    - Inclusions
    - Test Environments
    - Exclusions
  + Test Strategy
  + Defect Reporting Procedure
  + Roles/Responsibilities
  + Test Schedule
  + Test Deliverables
  + Pricing
  + Entry and Exit Criteria
  + Tools
  + Risks and Mitigations
  + Approvals

## **2.2 Use case, Test Scenario & Test Case**

------------------------------------------------------

* **Use Case:**
  + Use case describes the requirements.
  + Use case contains THREE items.
    - Actor, which is the user, which can be a single person or a group of people, interacting with a process.
    - Action, which is to reach the final outcome
    - Goat/Outcome, which is the successful user outcome.
* **Tests Scenario:**
  + A possible area to be tested (What to test)
* **Test Case:**
  + Step by step actions to be performed to validate functionality of AUT (how to test). AUT= Application Under Test
  + Test case contains test steps, expected result & actual result.

Testers will not write the use cases, product owner or project manager or who is the responsible for FRS document will write the use cases, which is the part of requirement.

Test Scenario is an area where we have to conduct testing. It basically describes the what to test, on the other hand Test case will describe how to test, use case describe describes part of requirements.

Test Scenario and Test Case are the Tester’s responsibility.

**Use case vs Test Case:**

--------------------------------

**Use case** – Describes functional requirement, prepared by Business Analyst (BA).

**Test Case** – Describes Test Steps / Procedure, prepared by Test Engineer.

**Test Scenario vs Test Case:**

--------------------------------

* Test Scenario is “What to be tested” and Test Case is “How to be Tested”.

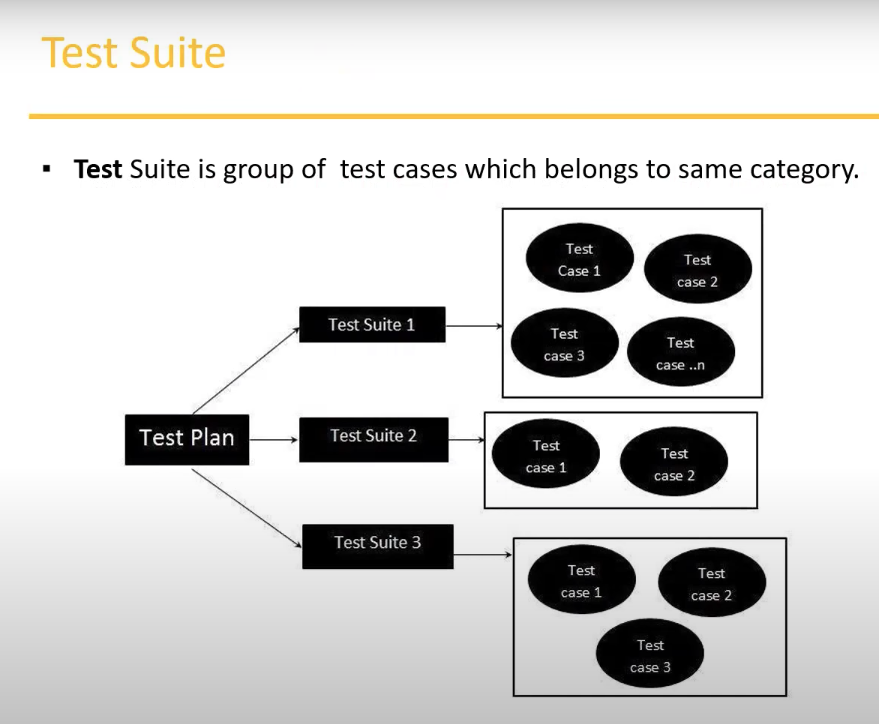
**Example:**

* Test Scenario: Checking the functionality of Login button. For this particular Test Scenario we can write these three Test cases.
  + TC1: Click the button without entering user name and password.
  + TC2: click the button only entering User name.
  + TC3: click the button while entering wrong user name and wrong password

## **2.3 Test Suit:**

--------------------------------

Test Suit is group of test cases which belongs to same category. For example here all the Sanity Test cases makes a group, it is called Sanity test suit. Or we can make a Regression Test suit etc.



**What is Test Case?**

**Ans.** A Test Case is a set of actions executed to validate particular feature of functionality of your software application.

**Test Case Contents**

---------------------------------

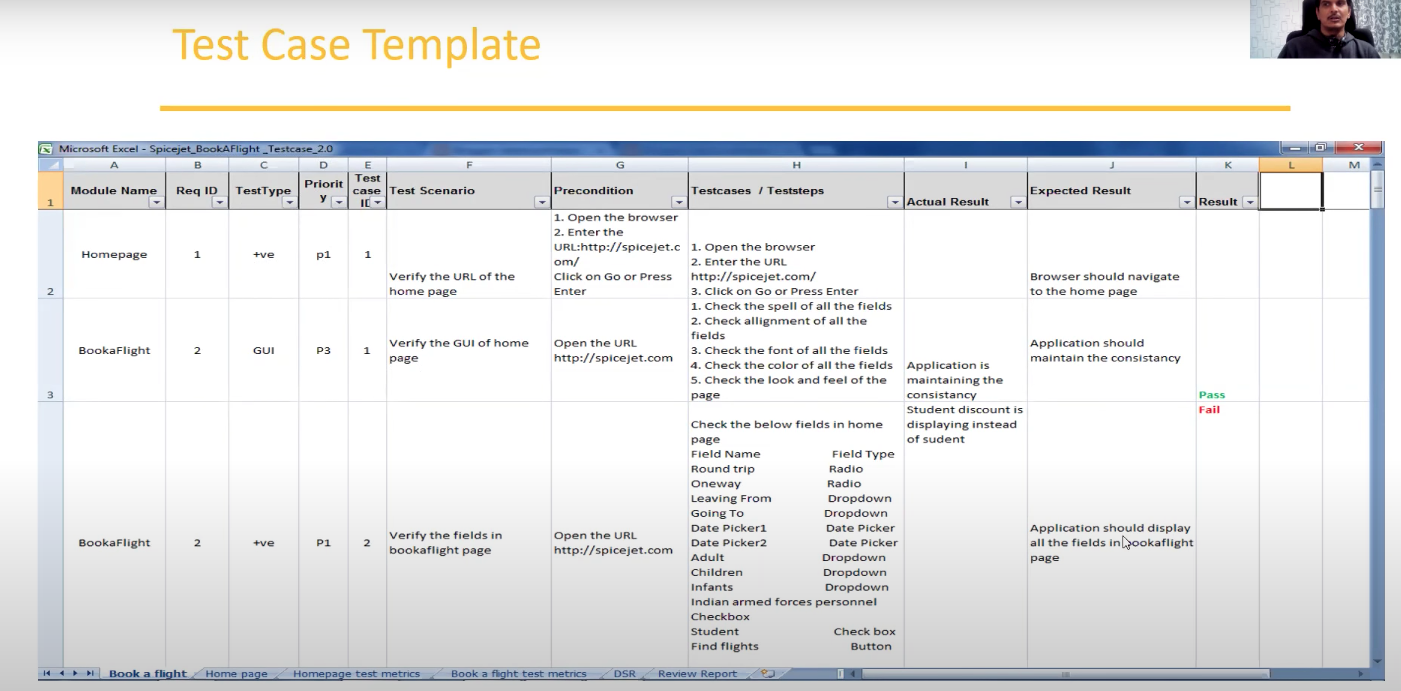
Test case is a document,

* Test case ID
* Test case title
* Description
* Pre-condition
* Priority (P0, P1, P2, P3) – other
* Requirement ID
* Steps/Actions
* Expected Results
* Actual Results
* Test Data

**Test Case Template**

--------------------------------

It could be deferent from company to company. These items should be there. Excel file is comfortable.



### **2.3.1 Requirement Traceability Matrix (RTM)**

---------------------------------------------------------

**Q. What is RTM (Requirement Traceability Matrix)?**

Ans.

Tester will prepare this while writing the test cases this

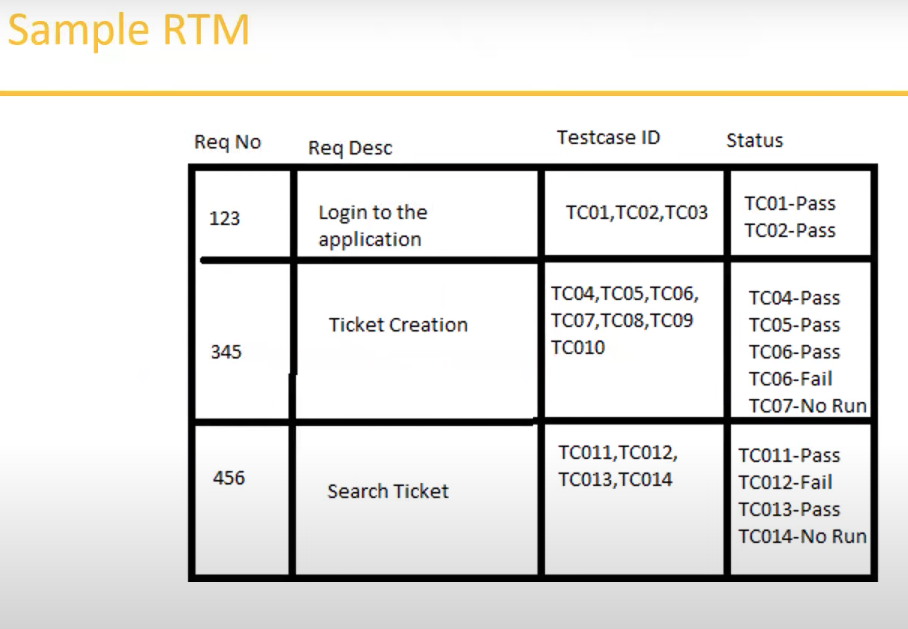
* + RTM describes the mapping of Requirement’s with the Test cases.
  + The main purpose of RTM is to see that all test cases are covered so that no functionality should miss while doing Software testing.

**Q. What parameters includes in the Traceability Matrix?**

Ans.

* + Requirement ID
  + Requirement Description
  + Test case ID’s

**Sample of RTM**



### **2.3.2 Test Environment:**

-----------------------------

* Test Environment is a platform specially build for test case execution on the software product.
* It is created by integrating the required software and hardware along with proper network configurations.
* Test environment simulates production/real time environment.
* Another name of test environment is Test Bed.

**Test Bed**: It is a collection of software and hardware environment we can create to perform the testing.

### **2.3.3 Test Execution:**

-----------------------

* During this phase test team will carry out the testing based on the test plans and the test cases prepared.
* **Entry Criteria**: Test cases, Test Data & Test Plan
* **Activities:**
  + Test cases are executed based on the test planning.
  + Status of test cases are marked, like Passed, Failed, Blocked, Run and others.
  + Documentation of test results and log defects for failed cases is done.
  + All the blocked and failed test cases are assigned bug ids.
  + Retesting once the defects are fixed.
  + Defects are tracked till closure.
* **Deliverables**: Provides defect and test cases execution report with completed results.

### **2.3.4 Guideline for Test Execution**

------------------------------------------

* The build being deployed to the QA environment is the most important part of the test execution cycle.
* Test execution is done in Quality Assurance(QA) environment.
* Test execution happens in multiple cycles.
* Test execution phase consists Executing the test cases + test scripts (if automation).

**Defects / Bugs**

-------------------------

* Any mismatched functionality found in a application is called as Defect/Bug/Issue.
* During Test Execution Test engineers are reporting mismatches as defects to develpers through templates or using tools.
* Defect Reporting Tools:
  + Clear Quest
  + DevTrack
  + Jira (This test management tool)
  + Quality Center (Test management tool)
  + Bug Jilla etc.
  + ALM

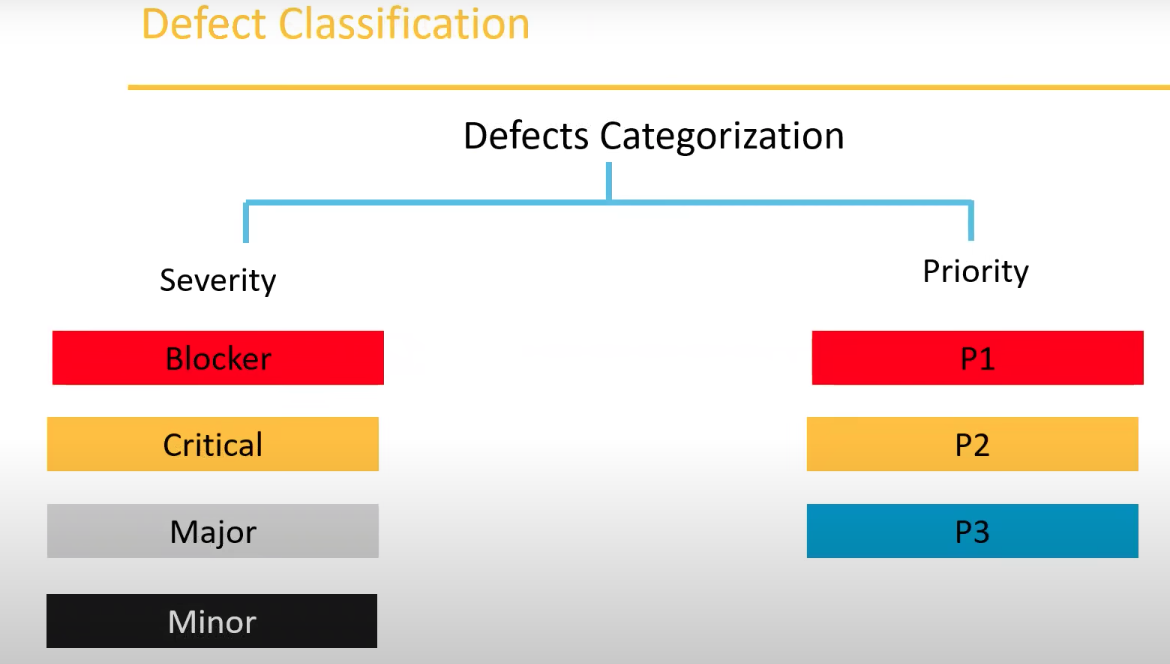
Bug tracking tools and Bug reporting tools are different. Defect tracking tools can only report the defects but the test management tools can report the bug and other activity also.

By the test management tool we can track each and every activity during the test.

**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 you provide reference to the documents like; requirements, design, architecture or may be even screenshots of the error to help understand the defect.
* **Detected By**: Name/ID of the developer who raised the defect.
* **Status**: Status of the defect, more on this later.
* **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.



### **2.3.5 Defect Severity**

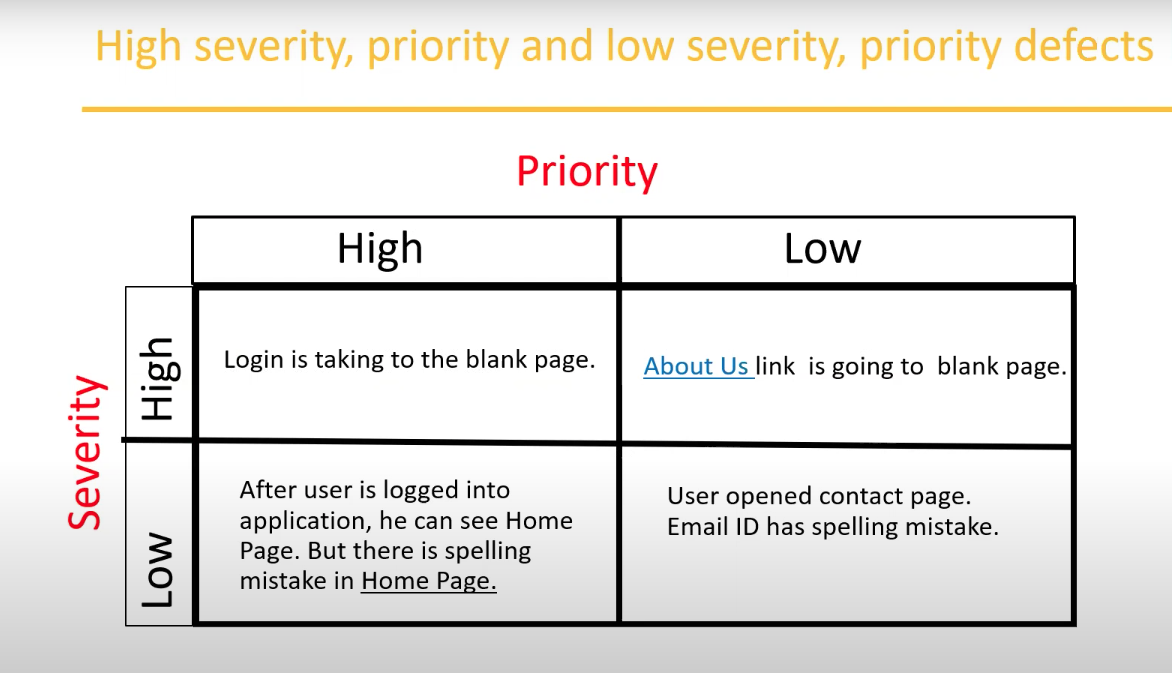
------------------------

* Severity describes the seriousness of defect and how much impact on Business workflow.
* Defect severity can be categorized into four class
  + **Blocker (Show stopper):** This Defect indicates nothing can proceed further.
    - Example: Application crashed, Login Not Worked
  + **Critical**: The main/basic functionality is not working. Customer business workflow is broken. They cannot proceed further.
    - Example 1: Fund transfer is not working in net banking
    - Example2: Ordering product in ecommerce application is not working.
  + **Major**: It cause some understandable behavior, but the feature/application is still functional.
    - Example1: After sending email there is no confirm message.
    - Example2: After booking cab there is no confirmation.
  + **Minor**: It won’t cause any major break-down of the system
    - Example: Look and feel issues, spellings, alignments.

### **2.3.6 Defect Priority**

--------------------------

* Priority describes the importance of defect.
* Defect Priority states the order in which a defect should be fixed.
* Defect priority can be categorized into three class
  + **P0 (High)** : The defect must be resolved immediately as it affects the system severely and cannot be used until it is fixed.
  + **P1 (Medium):** It can wait until a new versions/builds is created.
  + **P2 (Low):** Developer can fix it in later releases.



**More Examples:**

----------------------------------

* **Low priority-Low severity** – A spelling mistake in a page not frequently navigated by users.
* **Low Priority-High Severity** – Application crashing in some very corner case.
* **High priority – low severity** – Slight change in logo color or spelling mistake in company name.
* **High priority – High severity** – Issue with login functionality. (User is not able to login to the application).
* **High Severity – Low Priority** – Web page not found when user clicks on a link (user does not visit that page generally).
* **Low Priority – Low Severity** – Any cosmetic or spelling issues which is within a paragraph or in the page.

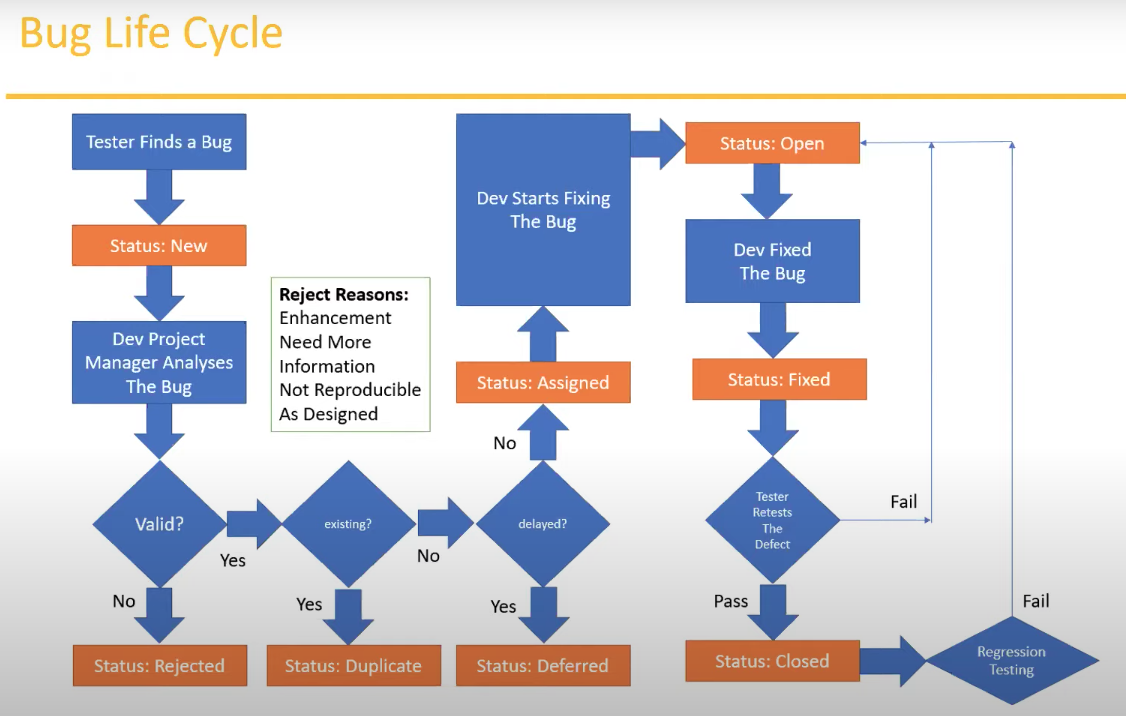
### **2.3.7 Defect Resolution:**

----------------------------

* 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 (This is a new feature, that will come in the next version)
  + Need more information
  + Not Reproducible
  + Fixed
  + As Designed

**Video # 9 (Last video of SDLC and STLC)**

## **2.4 Bug Life Cycle:**



## **2.5 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 summary 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

### **2.5.1 Test Metrics**

----------------------

We have to collect these data to calculate the Test Metrics

|  |  |
| --- | --- |
| Sl No. | Required Data |
| 1 | No. Of Requirements |
| 2 | Avg. No. of Test Cases Written Per Requirement |
| 3 | Total No. of Test Cases written for all Requirement |
| 4 | Total No. of Test cases Executed |
| 5 | No. of Test Cases Passed |
| 6 | No. of Test Cases Failed |
| 7 | No. of Test Cases Blocked |
| 8 | No. of Test Cases Unexecuted |
| 9 | Total no. of Defects identified |
| 10 | Critical Defects Count |
| 11 | Higher Defects Count |
| 12 | Medium Defects Count |
| 13 | Low Defects Count |
| 14 | Customer Defects |
| 15 | No. of Defects found in UAT |

By collecting these data we have to calculate these data:

* % of Test Cases Executed:
* % of Test cases Not Executed:
* % of Test Cases Passed:
* % of Test Cases Failed:
* % of Test Cases Blocked:

Defect Related Metrics:

* Defect Density
* Defect Removal Efficiency (DRE):

A = Defects identified during testing (Fixed Defects)

B = Defects identified by the customer (Missed defects)

* Defect Leakage:
* Defect Rejection Ratio:
* Defect Age:

Fixed date – Reported date

* Customer Satisfaction:

No. of complaints per period of time.

* + - * Time could be 1 month or 3 months.

### **2.5.2 QA / Testing Activities**

-------------------------------------

As a QA or QA Engineer these are the activities need to do.

* Understanding the requirements and functional specifications of the application.
* Identifying required Test Scenario’s
* Designing Test Cases to validate application.
* Setting up Test Environment (Test Bed)
* Execute Test Cases to valid application
* Log Test results (How many test cases pass/fail).
* Defect reporting and tracking.
* Retest fixed defects of previous build
* Perform various types of testing’s in application.
* Reports to Test Lead about the status of assigned tasks.
* Participated in regular team meetings.
* Creating automation scripts.
* Provides recommendation on whether or not the application / system is ready for production.

### **2.5.3 7 Principles of Software Testing:**

---------------------------------------------------

1. Start software testing at early stages. Means from the beginning when you get the requirements.
2. Test the software in order to find the defects.
3. Highly impossible to give the bug free software to the customer.
4. Should not do Exhaustive testing. Means we should not use same type of data for testing every time.
5. Testing is context based. Means decide what types of testing should be conducted based on type of application.
6. We should follow the concept of Pesticide Paradox. Means, if you are executing same cases for longer run, they wont be find any defects. We have to keep update test cases in every cycle/ release in order to find more defects.
7. We should follow defect clustering. Means some of the modules contains most of the defects. By experience, we can identify such risky modules. 80% of the problems are found in 20% of the modules.

**Video # 1**

Manual Testing Project

When we want to start a Manual Test Project, we have to start these steps sequentially.

* Project introduction
* Understanding & Explore the Functionality.
* Estimation
* Test Plan
* Writing Test Scenarios
* Writing Test Cases & Reviews
* Environment Setup & Build deployment
* Test Execution
* Bug Reporting & Tracking
* Sanity Testing, Re-Testing & Regression Testing
* Test Sign off.

Q: What is the difference between Project and Product?

Ans: Project is used for a particular customer as per their requirements and Product is used for the requirements of Customer from the market. That means a product can be used by the many users from the market but Project is used by a particular customer.

In the Software Testing we are using the Product.

1. **Project Introduction**

---------------------------------

What kind of Product it is?

It is an eCommerce product/Application

For the product Domain knowledge is very important. Example of Domains: Banking Domain, Educational Domain, eCommerce Domain etc.

For an eCommerce Application or Product there are some common activities such as:

* + Login
  + Search the products/items
  + Add these to Cart
  + Do Payment
  + Product will be delivered
  + Return the product etc.

There are 2 parts of an Application, one is Front End and the other is Bank End.

**Front end:** What activities are done by the customer is called Front end. Example: Login, Search the products/items, Add these to Cart etc. Front end activities is also called public activities through internet. It is also called **Internet Application**.

Back end: In this part only the admin can access. Add the product in the UI, Receive the money, send the product etc. in there environment. It is called **Intranet Application**.

As per the manual testing we are going to the Front end application.

Before getting the Application or Application link, we have to check the requirements in FRS.

The product link is: <https://www.opencart.com/index.php?route=cms/demo>

Here the customer is: Opencart.

1. **Understanding & Explore Functionality:**

-----------------------------------------------------------------

In the practical field they will not provide any UI just only they provide the documents.

Q: How will we understand the requirement of Application?

Ans: From the FRS (Functional Requirement Specification) documents.

In the FRS maybe the customer provides some Mockup Screens.

Q: What is Mockup screen?

Ans: These are some photos that will come before the application launch.

We will get plenty of time to read thoroughly the FRS document, highlight the important parts and note that. Because this is the base of Testing. If this is not prepare correctly, everything will not correct.

1. **Estimation**

------------------

There are something called Test Estimation, depends upon the number of functionality, complexity and number of Testers, we have to Estimate the deployment date. In this part it will estimate the total time will need for each part of the development.

1. **Test Plan**

------------------

Before writing the Test Plan we have to get some input, we will refer to the FRS or project plan that will be given by the management. How much time should be spent for testing, that is mentioned in project plan.

This is a document where the total information will write about scope of Testing, what to Test and what not to test, feature what should to be tested, what feature should not to test, schedule, what tools we have to use, what kind of testing we have to conduct, who will conduct, when will conduct. All of these information keep in one document is called Test Plan document.

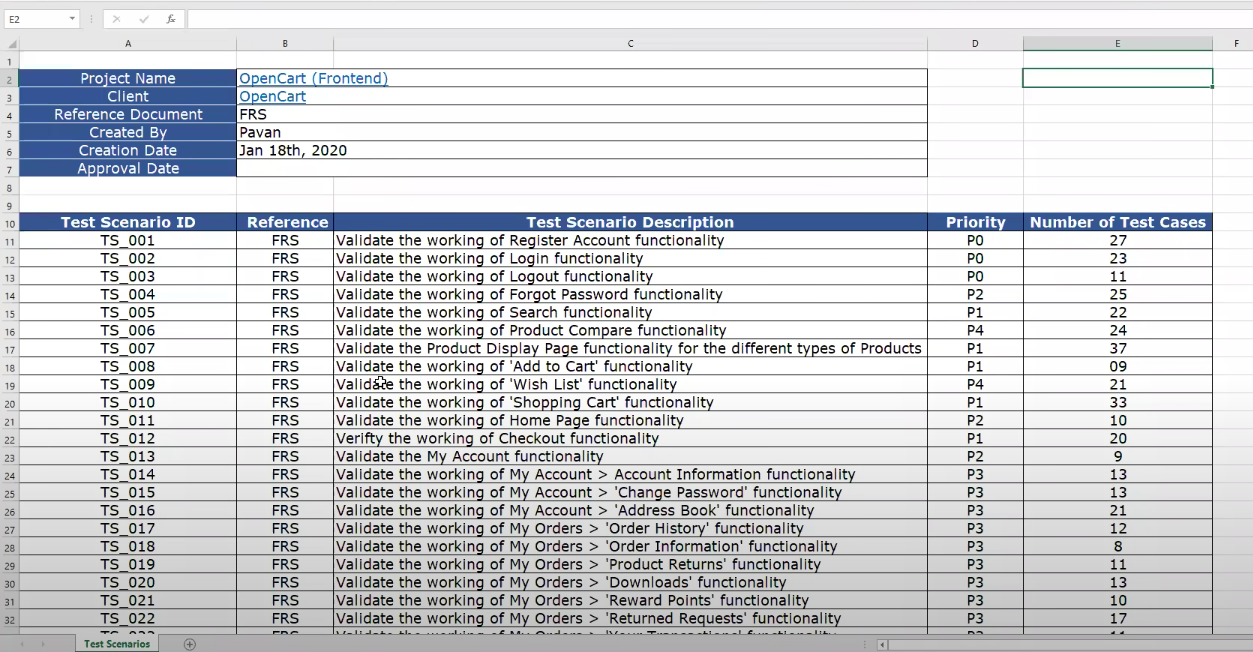
Normally Test Lead will prepare the Test Plan along with the TE.

1. **Test Scenarios**

---------------------------------

Test scenario is nothing but what to Test, area to be Tested. From the Scenarios we will derived the Test Cases. There are features available in the Scope, like: Login & Logout, Forget Password, Add to Cart etc. Each features have n numbers of Scenarios and n numbers of Test Cases.

Test Scenarios Example:



Priority P0 is the first priority, then P1, P2 and P3.

In this example we can see that For a Test Scenario TS 001, we have written 27 Test Cases.

“Validate” or “Verify” should use before the description of Test Scenarios.

How to write 27 Test Cases under the Test Scenario TS 001,

When we have observed in the register field, then we can see the various fields need to fill. Such as:

For Case 1: Fill all the mandatory field, then it register or not

For Case 2: Fill some of the mandatory field, then it register or not

For Case 3: Fill the email address with normal Text

For Case 4: Fill the password in different combination etc…..

**Video # 2 of the Project for Manual Testing**

1. **Test Scenarios**

---------------------------------

Every document should use the Version History, this will be in the First page.

For each scenario we can create one Excel sheet. For 31 Scenario, we can create 31 sheet in one excel file.

Rest of the video is for writing Test Cases, separate Excel file is available for that.

Video # 3 is for Environment setup

Video # 4 is for Test Execution.

For starting the Test execution, the prerequisite is the Test cases should be written and get approval from the management. Smoke testing and Sanity testing should pass. If these tests pass then we can accept the build, if these tests not passed then we will not accept the build.

We should follow the Test Plan, i.e. in which date we will start the test, when will finish, what is the size of team, how many cycle we defined etc. Test execution will happen in multiple cycles not only one or two cycles.

While testing the software if result mismatched between actual result and expected result, then we can consider as a bug, after then we can report to the developer. This bug reporting also has multiple ways, sometimes various companies uses some excel files or templates. Normally everyday we can send the bug report to the developer. Now a days, companies are using the bug tracking tools,

In Jira we can track all activities by using this tools.

The Test case file which made before we can update the status of Result and specify the Bug id.

There is a template for the Bug Report also same as Test Case file.

If any Bug is found, not to send the bug directly, try to test at least 3 times, and try on other machine also, if it exists after then we can report.

Severity of a Bug cannot change but the priority can be changed time to time.

Severity is impact on business, and priority is the how fast we have solve this.

After the Test Execution we have Bug Reporting & Tracking

Then Sanity Testing, Re-Testing & Regression Testing

After the meeting, then proceed for Test Sign off.

**Agile Model and Jira Tool**

What is Agile

Advantage & Disadvantages

Agile model / Agile Methodology / Agile Process

Q: **What is Agile?**

Ans: Agile is an Iterative and Incremental Model. Iterative means the same kind of process we are repeating again and again. Incremental means, at the beginning of the process we have a few number of features but in every cycle we will add new features.

In this process, customer no need to wait for the long time, it can be delivered piece by piece.

We can accommodate the new requirements.

In other words, It is a process model. It is a process with iterative and incremental approaches.

**Agile Principles**

--------------------------

1. No need to wait for long time
2. We develop, test and release a piece of software to the customer with few number of features.
3. We can accept / accommodate requirement changes.

There will be good communication between Customer, Business Analyst, Developers and Testers. Everybody works in a team. Delivery is very faster.

Advantages of Agile Process

----------------------------------------

1. Requirement changes are allowed in any stage of development (or) we can accommodate Requirement changes in the middle of development.
2. Releases will be very fast (Weekly)
3. Customer no need to wait for long time.
4. Good Communication between team.
5. It is very easy model to adopt.

Disadvantages of Agile process

-------------------------------------------------

Less focus on design and documentation since we deliver software very faster. Very less documents, most of time discuss in the meeting.

**Scrum**

-----------------

Sometimes maybe in the interview people can ask what is the difference between Agile and Scrum?

Ans.: Agile is a defined process, we have to follow like this and Scrum says how to follow them and how to implement those principles. Scrum is a kind of framework which will help us to develop and test the software in Agile process.

In shortly we can say, Scrum is a framework through which we build software product by following Agile Principles.

In the Scrum there are setup people will be involved as a team, we can call that as an Agile team or Scrum team.

Scrum includes group of people called as Scrum team, Normally contains 5-9 members.

* Product owner
* Scrum master
* Development team and
* QA team

**Product owner**

-----------------------

Product owner is person who will always contact with the customer, get the inputs and requirements from the customer.

**Roles of product owner:**

* Define the features of the product
* Prioritize features according to market value
* Adjust features and priority every iteration, as needed
* Accept or reject work results.

**Scrum Master:**

The main role is facilitate and driving the agile process. Most of the meeting conducted by the Scrum master. He is not a developer, Tester or any management role.

**Developers and QA:**

Develop and Test the Software.

**Scrum Terminology**

-------------------------------------

**User Story**: A Feature / module in a software. It is derived from Epics.

**Epic**: Collection of multiple stories.

Product owner will describe the requirement in the form of User Story and Epic

**Product Backlog**: Contains list of user stories. Prepared by Product owner.

**Sprint/Iteration**: Period of time to complete the user stories, decided by the product owner and team, usually 2-4 weeks of time. In this period the developer and testing team will make ready a piece of software and delivered to the customer. How many stories will complete in this sprint or time, the team will decide.

**Sprint planning meeting**: Meeting conducts with the team to define what can be delivered in the sprint and duration. Two things will be in main focus,

1. How many stories will have in the backlog, how many stories we are going to develop and test during the sprint,

2. What is the duration of the sprint.

This is one day meeting. Whenever starts project, the first day there is a Sprint planning meeting with the team.

**Sprint backlog**: List of committed stories by Dev/QA for specific sprint. In each sprint the sprint backlog will change. This is a subset of product Backlog.

**Scrum meeting**: Meeting conducted by Scrum Master **everyday 15 minutes**. Called **Scrum call** / **Standup meeting.**

In scrum meeting there are 3 things to discuss:

What will do today?

What did Yesterday? and

Is there any Impediments/blockage/delay/challenge?

**Sprint Retrospective meeting**: This meeting will conduct once after completion of every Sprint. Product owner, Scrum master, Dev and QA will attend in this meeting. In this meeting the main discussion about 3 things:

What went wrong?

What went well? And

What is the improvement we can do next sprint.

**Story point**: Rough estimation of user stories, will be given by Dev & QA in the form of Fibonacci series.

Fibonacci series: 0 1 1 2 3 5 8 13….

1 Story point = 1 hour /1 day (6 hours) it depends upon the company to company.

Login -----🡪 Dev -> 5 , QA -> 3 total story point will 8 i.e. 8 hours/1 day.

Burndown Chart = Shows how much work remaining in the sprint. Maintained by the scrum master daily.

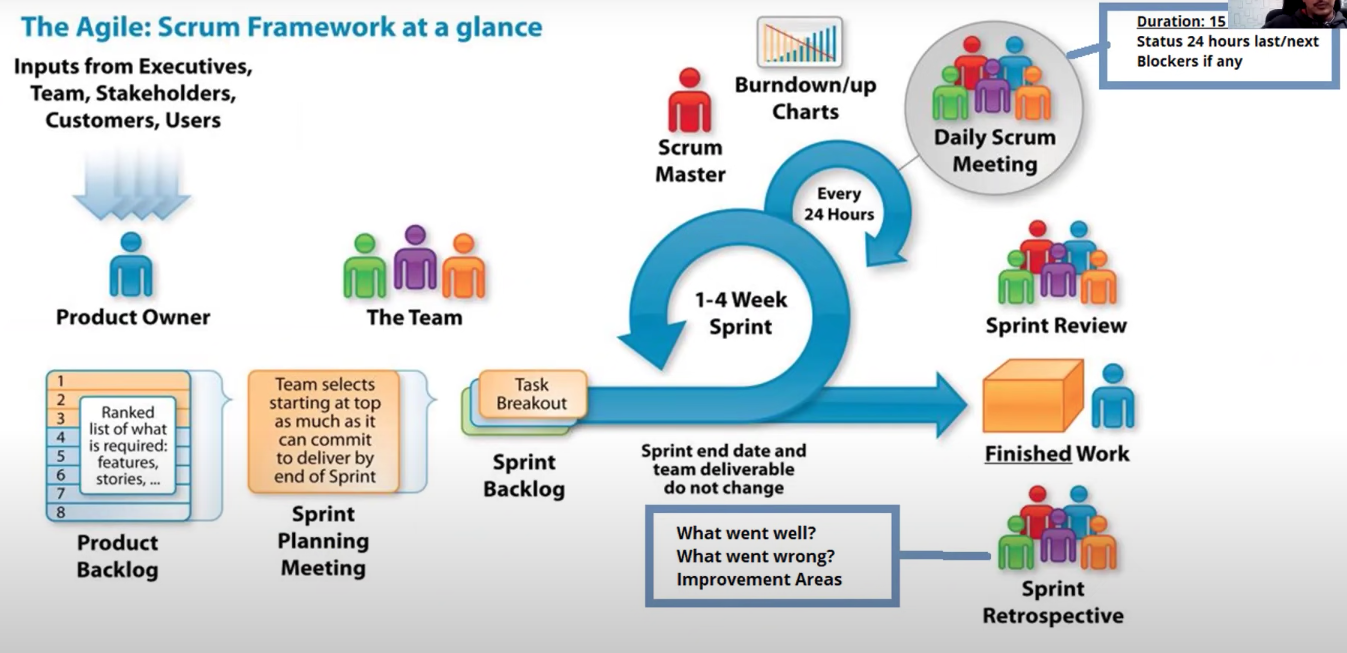


Fig: End to End process of Agile process

**Roles:**

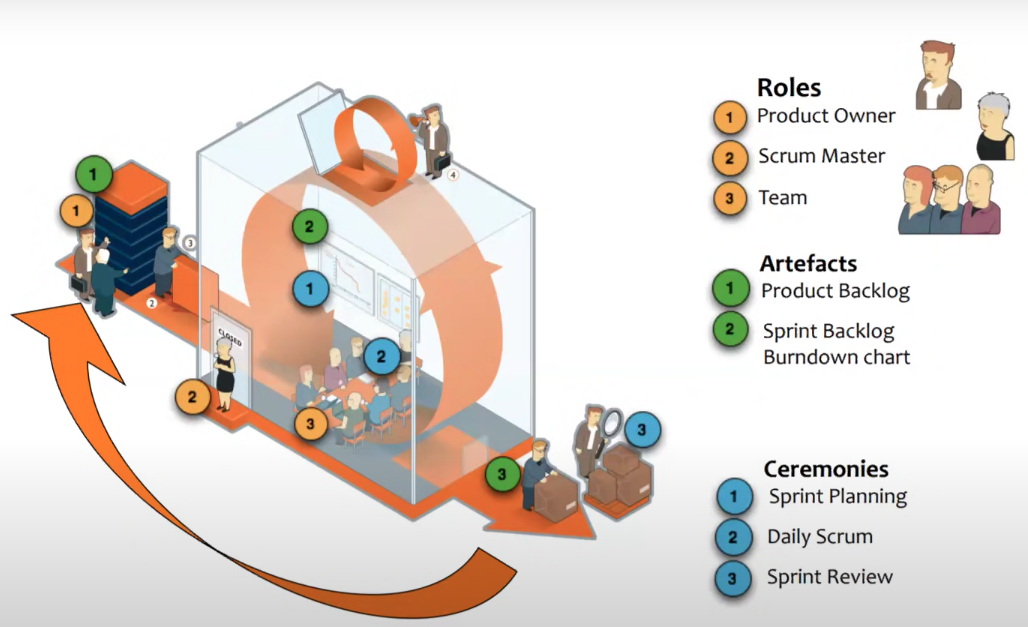
1. Product Owner
2. Scrum Master
3. Team

**Artefacts:**

1. Product Backlog
2. Sprint Backlog
3. Burndown chart

**Ceremonies:**

1. Sprint Planning
2. Daily Scrum
3. Sprint Review



Definition of Ready (DoR) & Definition of Done (DoD)

|  |  |
| --- | --- |
| **Definition of Ready (DoR)** | **Definition of Done (DoD)** |
| User Story is Clear | Code produced (all ‘to do’ items in code completed) |
| User Story is testable | Code commented, checked in and run against current version in source control |
| User Story is feasible | Peer reviewed (or produced with pair programming) and meeting development standards |
| User Story defined | Builds without errors |
| User Story Acceptance Criteria defined | Unit tests written and passing |
| User Story dependencies identified | Deployed to system test environment and passed system tests |
| User story sized by Development team | Passed UAT (user acceptance Testing) and signed off as meeting requirements |
| Scrum Team accepts User Experience artefacts | Any build / deployment /configuration changes are implemented /documented / communicated |
| Performance criteria identified, where appropriate | Relevant documentation / diagrams produced and / or updated |
| Team has a good idea what it will mean to Demo the User Story | Remaining hours for task set to zero and task closed |

Before writing the story, we have to confirm the DoR conditions. If these conditions fulfilled then we can say the Stories are ready.

When we can the Story is done? The Answers are the conditions mentioned in the DoD. All the Develop and Test works done, then we can say Definition of Done.

**Jira tool**

-------------------

There are multiple types of tools available in the software industry. Bug tracking tools, function automation tools, performance tools, security testing tools etc. specially for testing.

Management tools and automation tools

Whatever the activities doing throughout the testing life cycle, we are organizing the task, or managing the task, these come under test management tools.

Test Management tools: each and every activity can track in this tools like; bug reporting, writing test cases etc.

Bug Reporting & tracking tools:

Agile tools: This tool is designed for automate the Agile process. Specially to track the design task. Jira, VersionOne, Teamcity etc. are the examples of Agile tools.

Jira is a Agile Management tool.

Steps in Jira:

1. First step is to create a project. When first we created the instance at that time we provided the project name. So, by default, project name is there. If we need to create a new project, then we have to go the project menu and select the Create Project option. Then select classic project. Insert he project name, it will generate Project key.

If we go to Dashboard, then we will see all the information. The management people mostly they can see the information from here, what is the process progress, how many members in this project etc.

1. How to add users/people in Jira. Go to settings in the Right top corner, then the first option will navigate to management site in a new tab, where we can add users by adding the email address.

Backlog 🡪 Epic 🡪 Stories

Create sprint -🡪 Add stories to the sprint -🡪 Strat Sprint -🡪 add tasks for every story.

Click on Backlog, then to project, select the project, click on create version.

Give the version name (Example: OpenCart -V001). After then select the start date and finish date.

1. After then ‘Create Epic” will enable.

From the drop down we can select ‘Epic’, ‘Story’ or ‘Bug’. Provide the Short name, description etc. We can create multiple Epic at the same time.

1. How to create stories in Jira & add story points. To create stories umder this Epic, go to ‘Create’ button , then we have to select the ‘story’, then write the description and others. Product owner’s name will automatically come in the ‘Reporter’ field. If we want to select any Assignee then we can select from the list. We can provide the priority also. We can add multiple story.
2. Add the story points. Click on the story, then it will expand, after then we can put the story points.

After then in the daily planning meeting we have to decide which story will come in which sprint.

1. Creating sprint in Jira, Click on ‘Create Sprint’ then drag and drop the sprints to the box of sprint.
2. Adding user stories
3. Starting sprint, when the sprint added, then click on Start Sprint. Select the start and stop dates. Write some description.

During Scrum meeting, scrum master will get the information about the Sprint, then Scrum master will move the sprint from To do to done stage.

If any story is ‘Done’ stage it can now ready for Demo.

For a particular story we have to define the tasks, developer and QA has their own responsibility. So, we have to define the tasks for everyone. To add tasks in a story, Once we click on the story then a small screen will pop up. In this screen click on ‘Create subtask’. Here we can add multiple tasks. Example: understanding the requirements. Design, testing etc. Developer and tester will add the Tasks. Scrum master will move the task card from To-do to in progress, and done state.

If any story not completed within the sprint, we can add them in to the next sprint.

1. Adding sub-tasks to story
2. Sprint life cycle.

Adding Bug by selecting the ‘Bug’.

For Test Management activities we need to install ‘**Zepyre**’.

What is included in the Test Management activities?

Ans:

1. Writing Test Cases
2. Test Cycle (Regression test cycle, Sanity Test cycle etc.)
3. Update Test cases Passed/Failed/Blocked
4. Report the bugs
5. Generate Reports

Assignment from PIIT, Suvashish:

<https://magento.softwaretestingboard.com/>

This is a demo site.

Form this site, go to men section write an Epic, write the story and a task, find a bug from there.

1. Thursday night means at Friday After Tahajjud:
2. 11 times Istegfar
3. 11 times Durud Sharif
4. 11 times Ayatul Qursi
5. 11 times Durud Sharif
6. 1000 times Ya Samiu

After then Pray **Fajar Sunnat** and again

1. 11 times Durud Sharif
2. 11 times Ayatul Qursi
3. 11 times Durud Sharif

Then offer Faraj Salah, after then Dua (Monajat, whatever you need).

11 days Amal continue, need to must offer Tahajjud daily.

1. 3 days Amal (Thursday, Friday and Saturday)

After Faraj Salah of Fazar

1. 11 times Durud Sharif
2. 129 times Surah Kawsar
3. 11 times durd sharif

Fajar, Asar and Isha Everyday

-------------------------------------------------------------------------------------------------------------------------------

1. Thursday night means at Friday After Tahajjud:

1. 11 times Istegfar
2. 11 times Durud Sharif
3. 11 times Ayatul Qursi
4. 11 times Durud Sharif
5. 1000 times Ya Samiu

After then Pray **Fajar Sunnat** and again

1. 11 times Durud Sharif
2. 11 times Ayatul Qursi
3. 11 times Durud Sharif

Then offer Faraj Salah, after then Dua (Monajat, whatever you need).

1. ays Amal continue, need to must offer Tahajjud daily.

2. 3 days Amal (Thursday, Friday and Saturday)

After Faraj Salah of Fazar

1. 11 times Durud Sharif
2. 129 times Surah Kawsar
3. 11 times durd sharif

Fajar, Asar and Isha Everyday