# SOFTWARE TESTING

## INTRODUCTION :-

- **WHAT IS SDLC ?**

A Software Development Life Cycle is essentially a series of steps or phases that provide a model for the development and lifecycle management of an application or piece of software.

- **WHAT IS SOFTWARE TESTING ?**

Software Testing is a Process of executing a Program or Application with the intend of finding Software Bugs. It can also be stated as the process of validating and verifying that a software program or application or product.

- **WRITE SDLC PHASES WITH BASIC INTRODUCTION :-**

1) REQUIREMENT COLLECTION / GATHERING :-

Requirements gathering is a crucial phase in the software development life cycle . It involves collecting, documenting, and managing the requirements that define the features and functionalities of a system or application and also gathering information about the software requirements from stakeholders, such as customers, end-users, and business analysts.

Requirements Gathering stage need teams to get detailed and precise requirements. This helps companies to finalize the necessary timeline to finish the work of that system.

2) ANALYSIS PHASE :-

The analysis phase in SDLC is where a thorough examination is done to get a clear understanding of the business requirements as outlined in the Business Case and Scope documents. The customer's issue is outlined at this stage. At the conclusion of this stage, a requirement document will have been produced as the deliverable result.

The analysis stage includes gathering all the specific details required for a new system as well as determining the first ideas for prototypes.

Developers may:

Define any prototype system requirements

Evaluate alternatives to existing prototypes

Perform research and analysis to determine the needs of end-users

Furthermore, developers will often create a software requirement specification or SRS document.

This includes all the specifications for software, hardware, and network requirements for the system they plan to build. This will prevent them from overdrawing funding or resources when working at the same place as other development teams.

3) DESIGN PHASE :-

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

- High-Level Design (HLD) :-

Brief description and name of each module
An outline about the functionality of every module
Interface relationship and dependencies between modules
Database tables identified along with their key elements
Complete architecture diagrams along with technology details

- Low-Level Design (LLD) :-

Functional logic of the modules
Database tables, which include type and size
Complete detail of the interface
Addresses all types of dependency issues
Listing of error messages
Complete input and outputs for every module

4) IMPLEMENTATION PHASE :-

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

5) TESTING PHASE :-

Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement.

During this phase, QA and testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system.

6) MAINTENANCE :-

Once the system is deployed, and customers start using the developed system, following 3 activities occur

Bug fixing – bugs are reported because of some scenarios which are not tested at all

Upgrade – Upgrading the application to the newer versions of the Software

Enhancement – Adding some new features into the existing software

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

## ❖ EXPLAIN PHASES OF WATERFALL MODEL :-

Waterfall Model is a classical software development methodology that was first introduced by Winston W. Royce in 1970. It is a linear and sequential approach to software development that consists of several phases that must be completed in a specific order.

The Waterfall Model has six phases:

1. **Requirements Gathering and Analysis:** The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope and objectives of the project.

**2. Design Phase:** Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.

**3. Implementation and Unit Testing:** The implementation phase involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.

**4. Integration and System Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.

**5. Deployment:** Once the software has been tested and approved, it is deployed to the production environment.

**6. Maintenance:** The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.

The classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after the completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus the development process can be considered as a sequential flow in the waterfall. Here the phases do not overlap with each other

## ❖ WRITE PHASES OF SPIRAL MODEL :-

It has four stages or phases: The planning of objectives, risk analysis, engineering or development, and finally review. A project passes through all these stages repeatedly and the phases are known as a Spiral in the model.

**Determine objectives and find alternate solutions –** This phase includes requirement gathering and analysis. Based on the requirements, objectives are defined and different alternate solutions are proposed.

**Risk Analysis and resolving –** In this quadrant, all the proposed solutions are analyzed and any potential risk is identified, analyzed, and resolved.

**Develop and test:** This phase includes the actual implementation of the different features. All the implemented features are then verified with thorough testing.

**Review and planning of the next phase –** In this phase, the software is evaluated by the customer. It also includes risk identification and monitoring like cost overrun or schedule slippage and after that planning of the next phase is started.

## ❖ WHAT IS AGILE METHODOLOGY :-

Agile methodology is a project management framework that breaks projects down into several dynamic phases, commonly known as sprints.

The Agile framework is an iterative methodology. After every sprint, teams reflect and look back to see if there was anything that could be improved so they can adjust their strategy for the next sprint.

- **WRITE AGILE MANIFESTO PRINCIPLES :-**

The Agile Manifesto is a document that focuses on four values and 12 principles for Agile software development.

The 12 principles used in Agile methodology are:

1. Satisfy customers through early, continuous improvement and delivery.When customers receive new updates regularly, they're more likely to see the changes they want within the product. This leads to happier, more satisfied customers—and more recurring revenue.
2. Welcome changing requirements, even late in the project. The Agile framework is all about adaptability. In iterative processes like Agile, being inflexible causes more harm than good.
3. Deliver value frequently. Similar to principle #1, delivering value to your customers or stakeholders frequently makes it less likely for them to churn.
4. Break the silos of your projects. Collaboration is key in the Agile framework. The goal is for people to break out of their own individual projects and collaborate together more frequently.
5. Build projects around motivated individuals. Agile works best when teams are committed and actively working to achieve a goal.
6. The most effective way to communicate is face-to-face. If you're working on a distributed team, spend time communicating in ways that involve face-to-face communication like Zoom calls.
7. Working software is the primary measure of progress. The most important thing that teams should strive for with the Agile framework is the product. The goal here is to prioritize functional software over everything else.
8. Maintain a sustainable working pace. Some aspects of Agile can be fast-paced, but it shouldn't be so fast that team members burn out. The goal is to maintain sustainability throughout the project.
9. Continuous excellence enhances agility. If the team develops excellent code in one sprint, they can continue to build off of it the next. Continually creating great work allows teams to move faster in the future.
10. Simplicity is essential. Sometimes the simplest solution is the best solution. Agile aims to not overcomplicate things and find simple answers to complex problems.
11. Self-organizing teams generate the most value. Similar to principle #5, proactive teams become valuable assets to the company as they strive to deliver value.
12. Regularly reflect and adjust your way of work to boost effectiveness. Retrospective meetings are a common Agile practice. It's a dedicated time for teams to look back and reflect on their performance and adapt their behaviors for the future.

- **EXPLAIN WORKING METHODOLOGY OF AGILE MODEL AND ALSO WRITE PROS AND CONS :-**

Agile Methodology meaning a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project. In the Agile model in software testing, both development and testing activities are concurrent, unlike the Waterfall model.

Agile methodology is a project management framework that breaks projects down into several dynamic phases, commonly known as sprints. The Agile framework is an iterative methodology. After every sprint, teams reflect and look back to see if there was anything that could be improved so they can adjust their strategy for the next sprint.

- **PROS :-**

1. In Agile methodology the delivery of software is unremitting.
2. The customers are satisfied because after every Sprint working feature of the software is delivered to them.
3. Customers can have a look of the working feature which fulfilled their expectations.

4. If the customers has any feedback or any change in the feature then it can be accommodated in the current release of the product.
5. In Agile methodology the daily interactions are required between the business people and the developers.
6. In this methodology attention is paid to the good design of the product.
7. Changes in the requirements are accepted even in the later stages of the development.
8. An Agile/Scrum approach can improve organizational synergy by breaking down organizational barriers and developing a spirit of trust and partnership around organizational goals.

- **CONS :-**

1. In Agile methodology the documentation is less.
2. Sometimes in Agile methodology the requirement is not very clear hence it's difficult to predict the expected result.
3. In few of the projects at the starting of the software development life cycle it's difficult to estimate the actual effort required.
4. Because of the ever-evolving features, there is always a risk of the ever-lasting project.
5. For complex projects, the resource requirement and effort are difficult to estimate.

❖ **WHAT IS SRS :-**

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill the needs of all stakeholders (business, users).

❖ **WHAT IS OOPS :-**

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior.

Object-Oriented Programming System (OOPs) is a programming concept that works on the principles of abstraction, encapsulation, inheritance, and polymorphism. It allows users to create objects they want and create methods to handle those objects. The basic concept of OOPs is to create objects, re-use them throughout the program, and manipulate these objects to get results.

❖ **WRITE BASIC CONCEPTS OF OOPS :-**

1) Class

The class is one of the Basic concepts of OOPs which is a group of similar entities. It is only a logical component and not the physical entity. Lets understand this one of the OOPs Concepts with example, if you had a class called "Expensive Cars" it could have objects like Mercedes, BMW, Toyota, etc. Its properties(data) can be price or speed of these cars. While the methods may be performed with these cars are driving, reverse, braking etc.

2) Object

An object can be defined as an instance of a class, and there can be multiple instances of a class in a program. An Object is one of the Java OOPs concepts which contains both the data and the function, which operates on the data. For example – chair, bike, marker, pen, table, car, etc.

3) Inheritance

Inheritance is one of the Basic Concepts of OOPs in which one object acquires the properties and behaviors of the parent object. It's creating a parent-child relationship between two classes. It offers robust and natural mechanism for organizing and structure of any softwar.

4) Polymorphism

Polymorphism refers to one of the OOPs concepts which is the ability of a variable, object or function to take on multiple forms. For example, in English, the verb *run* has a different meaning if you use it with *a laptop*, *a foot race*, and *business*. Here, we understand the meaning of *run* based on the other words used along with it. The same also applied to Polymorphism.

5) Abstraction

Abstraction is one of the OOP Concepts which is an act of representing essential features without including background details. It is a technique of creating a new data type that is suited for a specific application. Lets understand this one of the OOPs Concepts with example, while driving a car, you do not have to be concerned with its internal working. Here you just need to concern about parts like steering wheel, Gears, accelerator, etc.

6) Encapsulation

Encapsulation is one of the best OOPs concepts of wrapping the data and code. In this OOPs concept, the variables of a class are always hidden from other classes. It can only be accessed using the methods of their current class. For example – in school, a student cannot exist without a class.

- **WHAT IS OBJECT :-**

Object is an instance of a class. An object in OOPS is nothing but a self-contained component which consists of methods and properties to make a particular type of data useful. For example color name, table, bag, barking. When you send a message to an object, you are asking the object to invoke or execute one of its methods as defined in the class.

From a programming point of view, an object in OOPS can include a data structure, a variable, or a function. It has a memory location allocated. Java Objects are designed as class hierarchies.

- **WHAT IS CLASS :-**

Class are a blueprint or a set of instructions to build a specific type of object. It is a basic concept of Object-Oriented Programming which revolve around the real-life entities. Class in Java determines how an object will behave and what the object will contain.

- **WHAT IS ENCAPSULATION :-**

Encapsulation is a mechanism to wrap up variables(data) and methods(code) together as a single unit. It is the process of hiding information details and protecting data and behavior of the object. It is one of the four important OOP concepts. The encapsulate class is easy to test, so it is also better for unit testing.

- **WHAT IS INHERITANCE :-**

Inheritance is a mechanism in which one class acquires the property of another class. For example, a child inherits the traits of his/her parents. With inheritance, we can reuse the fields and methods of the existing class. Hence, inheritance facilitates Reusability and is an important concept of OOPs.

- **WHAT IS POLYMORPHISM:-**

Polymorphism occurs when there are one or more classes or objects related to each other by inheritance. It is the ability of an object to take many forms. Inheritance lets users inherit attributes and methods, and polymorphism uses these methods to perform different tasks. So, the goal is communication, but the approach is different.