## Software Testing Assignment

## Module–1(Fundamental)

## What is SDLC

## **SDLC** is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software that meets customer expectations. The system development should be complete in the pre-defined time frame and cost. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life Cycle has its own process and deliverables that feed into the next phase. SDLC stands for **Software Development Life Cycle** and is also referred to as the Application Development life-cycle

## Write SDLC phases with basic introduction

## SDLC Phases

## 1: Requirement collection and analysis

## 2: Feasibility study

## 3: Design

## 4: Coding

## 5: Testing

## 6: Installation/Deployment

## 7: Maintenance

## 1: Requirement collection and analysis

## The requirement is the first stage in the SDLC process. It is conducted by the senior team members with inputs from all the stakeholders and domain experts in the industry. Planning for the [quality assurance](https://www.guru99.com/all-about-quality-assurance.html) requirements and recognization of the risks involved is also done at this stage.

## This stage gives a clearer picture of the scope of the entire project and the anticipated issues, opportunities, and directives which triggered the project.

## 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: Feasibility study

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

## **There are mainly five types of feasibilities checks:**

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

## **Legal:** Can we handle this project as cyber law and other regulatory framework/compliances.

## **Operation feasibility:** Can we create operations which is expected by the client?

## **Technical:** Need to check whether the current computer system can support the software

## **Schedule:** Decide that the project can be completed within the given schedule or not.

## 3: Design

## In this third phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture.

## This design phase serves as input for the next phase of the model.

## There are two kinds of design documents developed in this phase:

## **High-Level Design (HLD)**

## Brief description and name of each module

## An outline about the functionality of every module

## Interface relationship and dependencies between modules

## Database tables identified along with their key elements

## Complete architecture diagrams along with technology details

## **Low-Level Design (LLD)**

## Functional logic of the modules

## Database tables, which include type and size

## Complete detail of the interface

## Addresses all types of dependency issues

## Listing of error messages

## Complete input and outputs for every module

## 4: Coding

## Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process.

## In this phase, Developer needs to follow certain predefined coding guidelines. They also need to use [programming tools](https://www.guru99.com/software-development-tools.html) like compiler, interpreters, debugger to generate and implement the code.

## 5: Testing

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

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

## 6: Installation/Deployment

## Once the software testing phase is over and no bugs or errors left in the system then the final deployment process starts. Based on the feedback given by the project manager, the final software is released and checked for deployment issues if any.

## 7: Maintenance

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

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

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

## Enhancement – Adding some new features into the existing software

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

## What is software testing?

## **Software Testing** is a method to check whether the actual software product matches expected requirements and to ensure that software product is[Defect](https://www.guru99.com/defect-management-process.html)free.

## It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest.

## The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

## What is agile methodology?

## The **Agile software development** methodology is one of the simplest and effective processes to turn a vision for a business need into software solutions.

## Agile is a term used to describe software development approaches that employ continual planning, learning, improvement, team collaboration, evolutionary development, and early delivery. It encourages flexible responses to change.

## Pros

## Is a very realistic approach to software development

## Promotes teamwork and cross training.

## Functionality can be developed rapidly and demonstrated.

## Resource requirements are minimum.

## Suitable for fixed or changing requirements

## Delivers early partial working solutions.

## Good model for environments that change steadily

## Minimal rules, documentation easily employed

## Enables concurrent development and delivery within an Overall planned context.

## Little or no planning required Easy to manage.

## Gives flexibility to developers

## Cons

## Not suitable for handling complex dependencies.

## More risk of sustainability, maintainability and extensibility.

## An overall plan, an agile leader and agile PM practice is a must without which it will not work.

## Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

## Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction. There is very high individual dependency, since there is minimum documentation generated.

## Transfer of technology to new team members may be quite challenging due to lack of documentation.

## 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).

## SRS is a blueprint or roadmap for the software you're going to build.

## The elements that comprise an SRS can be simply summarized into four Ds:

## [Define your product's purpose.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#purpose)

## [Describe what you're building.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#describe)

## [Detail the requirements.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#detail)

## [Deliver it for approval.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#approve)

## Explain Phases of the waterfall model

## **Waterfall Model** is a sequential model that divides software development into pre-defined phases. Each phase must be completed before the next phase can begin with no overlap between the phases. Each phase is designed for performing specific activity during the SDLC phase.

## Waterfall Model in SDLC

## Use of Waterfall Model

## Requirements are not changing frequently

## Application is not complicated and big

## Project is short

## Requirement is clear

## Environment is stable

## Technology and tools used are not dynamic and is stable

## Resources are available and trained

## Different phases

## Requirement Gathering stage

## During this phase, detailed requirements of the software system to be developed are gathered from client

## Design Stage

## Plan the programming language, for Example [Java](https://www.guru99.com/java-tutorial.html), [PHP](https://www.guru99.com/php-tutorials.html), .net or database like Oracle, MySQL, etc.Other high-level technical details of the project

## Implementation or Built Stage

## After design stage, it is Implementation or built stage, that is nothing but coding the software.

## Test Stage

## In this phase, you test the software to verify that it is built as per the specifications given by the client.

## Deployment stage

## Deploy the application in the respective environment

## Maintenance stage

## Once your system is ready to use, you may later require change the code as per customer request

## Write phases of spiral model

## Spiral Model is a risk-driven software development process model.

## It is a combination of waterfall model and iterative model. Spiral Model helps to adopt software development elements of multiple process models for the software project based on unique risk patterns ensuring efficient development process.

## The development process in Spiral model in SDLC, starts with a small set of requirement and goes through each development phase for those set of requirements. The software engineering team adds functionality for the additional requirement in every-increasing spirals until the application is ready for the production phase.

## What is Spiral Model in Software Engineering? Phases, Use, Advanatges & Disadvantages - Binary Terms

## Uses of Spiral model:

## When costs there are a budget constraint and risk evaluation is important.

## For medium to high-risk projects

## Long-term project commitment because of potential changes to economic priorities as the requirements change with time.

## Customer is not sure of their requirements which are usually the case

## Requirements are complex and need evaluation to get clarity

## New product line which should be released in phases to get enough customer feedback

## Significant changes are expected in the product during the development cycle.

## Pros

## Changing requirenents can be aCcommodated.

## Allows for extensive use of prototypes

## Requirements can be captured more accurately.

## Users see the system early.

## Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.

## Cons

## Management is more complex.

## End of project may not be known early.

## Not suitable for small or low risk projects and could be expensive for small projects.

## Process is complex

## Spiral may go indefinitely

## Large number of intermediate stages requires excessive documentation.

## Spiral Model Phases

## Planning

## It includes estimating the cost, schedule and resources for the iteration. It also involves understanding the system requirements for continuous communication between the system analyst and the customer

## Risk Analysis

## Identification of potential risk is done while risk mitigation strategy is planned and finalized

## **Engineering**

## It includes testing, coding and deploying software at the customer site

## **Evaluation**

## Evaluation of software by the customer also, includes identifying and monitoring risks such as schedule slippage and cost overrun

## Explain working methodology of agile model and also write pros and cons.

## Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes (small time frames) to deliver specific features for a release.

## Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

## Agile thought process had started early in the softwaredevelopment and started becoming popular with time due to its flexibility and adaptability.

## Pros

## Is a very realistic approach to software development

## Promotes teamwork and cross training.

## Functionality can be developed rapidly and demonstrated.

## Resource requirements are minimumn.

## Suitable for fixed or changing requirements

## Delivers early partial working slutions.

## Good model for environmentS that change steadily

## Minimal rules, documentation easily employed

## Enables concurrent development and delivery within an

## Overall

## planned context.

## Little or no planning required

## Easy to manage .

## Gives flexibility to developers

## Cons

## Not suitable for handling complex dependencies.

## More risk of sustainability, maintainability and extensibility.

## An overall plan, an agile leader and agile PM practice is a must without which it will not work.

## Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

## Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.

## There is very high individual dependency, since there is minimum documentation generated.

## Transfer of technology to new team members may be quite challenging due to lack of documentation.

## Draw uasecase for online book shopping

## .

## 

## Draw uasecase for online bill payment system

