**MODULE: 1 (SDLC)**

**• What is software? What is software engineering?**

Ans=Software is a collection of codes, documents, and triggers that does a specific job fills a specific requirement.

Software engineering is a systematic engineering approach to software development. A software engineering is a person who applies the principle of software engineering to design, develop, maintain, test, and evaluate computer software.

**• Explain types of software**

Ans=Application Software

This is the most common type of computer software, and can be defined as end-user programs that help you perform tasks or achieve a desired outcome. The end-user is the person who is actually using a product or program. (They are the one for whom the “end result” is designed.) Some examples of application software include internet browsers, a CRM tool like Hubspot, a photo-editing software like Adobe or Lightroom, or a word processing application like Microsoft Word. Application software is installed on a computer or mobile device based upon a user’s need. Because this is the most common type of software, there are many options available and users can choose the one that best fits their needs, budget, and expectations. (For example, anyone wanting to look on the internet could use Chrome, Safari, or even Firefox.)

System Software

System software helps the user, the computer or mobile device, and an application all work together seamlessly. This makes system software crucial to running any kind of application software as well as the whole computer system.

Think about when your laptop or phone has an update. This is system software in action: there is a tweak made to the system software that helps your computer or phone continue to work well and keep applications running. Apple’s iOS is an example of system software, as is Microsoft Windows. System software is always running in the background of your device, but it is never something you will use directly. In fact, the only time most people remember it’s there is when it is time for an update.

Programming Software

While application software is designed for end-users, and system software is designed for computers or mobile devices, programming software is for computer programmers and developers who are writing code. These are programs that are used to write, develop, test, and debug other software programs. It’s helpful to think of these programs as a translator of sorts: they take programming languages like Laravel, Python, C++, and more and translate them into something a computer or phone will understand.

**• What is SDLC? Explain each phase of SDLC**

An SDLC (software development life cycle) is **a big-picture breakdown of all the steps involved in software creation (planning, coding, testing, deploying, etc.)**.

7 Stages of the System Development Life Cycle

There are seven primary stages of the modern system development life cycle. Here’s a brief breakdown:

* Planning Stage
* Feasibility or Requirements of Analysis Stage
* Design and Prototyping Stage
* Software Development Stage
* Software Testing Stage
* Implementation and Integration
* Operations and Maintenance Stage

**Planning Stage**

* Before we even begin with the planning stage, the best tip we can give you is to take time and acquire proper [understanding of app development life cycle.](https://clouddefense.ai/blog/understanding-app-development-life-cycle)
* The planning stage (also called the feasibility stage) is exactly what it sounds like: the phase in which developers will plan for the upcoming project.
* It helps to define the problem and scope of any existing systems, as well as determine the objectives for their new systems.
* By developing an effective outline for the upcoming development cycle, they'll theoretically catch problems before they affect development.
* And help to secure the funding and resources they need to make their plan happen.
* Perhaps most importantly, the planning stage sets the project schedule, which can be of key importance if development is for a commercial product that must be sent to market by a certain time.

**Analysis Stage**

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.

**Design Stage**

The design stage is a necessary precursor to the main developer stage.

Developers will first outline the details for the overall application, alongside specific aspects, such as its:

* User interfaces
* System interfaces
* Network and network requirements
* Databases

They’ll typically turn the SRS document they created into a more logical structure that can later be implemented in a programming language. Operation, training, and maintenance plans will all be drawn up so that developers know what they need to do throughout every stage of the cycle moving forward.

Once complete, development managers will prepare a design document to be referenced throughout the next phases of the SDLC.

**Development Stage**

The development stage is the part where developers actually write code and build the application according to the earlier design documents and outlined specifications.

This is where [Static Application Security Testing](https://clouddefense.ai/sast-static-application-security-testing) or SAST tools come into play.

Product program code is built per the design document specifications. In theory, all of the prior planning and outlined should make the actual development phase relatively straightforward.

Developers will follow any coding guidelines as defined by the organization and utilize different tools such as compilers, debuggers, and interpreters.

Programming languages can include staples such as C++, PHP, and more. Developers will choose the right programming code to use based on the project specifications and requirements.

**Testing Stage**

Building software is not the end.

Now it must be tested to make sure that there aren’t any bugs and that the end-user experience will not negatively be affected at any point.

During the testing stage, developers will go over their software with a fine-tooth comb, noting any bugs or defects that need to be tracked, fixed, and later retested.

t’s important that the software overall ends up meeting the quality standards that were previously defined in the SRS document.

Depending on the skill of the developers, the complexity of the software, and the requirements for the end-user, testing can either be an extremely short phase or take a very long time. Take a look at our [top 10 best practices for software testing projects](https://clouddefense.ai/blog/10-best-practices-for-software-testing-projects) for more information.

### Implementation and Integration Stage

After testing, the overall design for the software will come together. Different modules or designs will be integrated into the primary source code through developer efforts, usually by leveraging training environments to detect further errors or defects.

The information system will be integrated into its environment and eventually installed. After passing this stage, the software is theoretically ready for market and may be provided to any end-users.

**Maintenance Stage**

The SDLC doesn’t end when software reaches the market. Developers must now move into a maintenance mode and begin practicing any activities required to handle issues reported by end-users.

Furthermore, developers are responsible for implementing any changes that the software might need after deployment.

This can include handling residual bugs that were not able to be patched before launch or resolving new issues that crop up due to user reports. Larger systems may require longer maintenance stages compared to smaller systems.

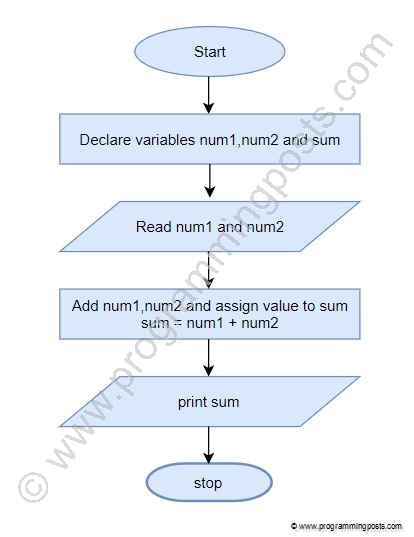
**• What is DFD? Create a DFD diagram on Flipkart**

**DFD** is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modeling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.

**• What is Flow chart? Create a flowchart to make addition of two numbers**

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows.

**Flowchart :**



**• What is Use case Diagram? Create a use-case on bill payment on paytm.**

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

‍