Q1. What is software? What is software engineering?

Ans. Software engineering is a concept in and of itself, but to better understand it, you need to know what each part of the term means before you can fully understand how they operate together. It can be difficult to understand, even though it does seem straightforward. That is because the pieces are more complicated than many believe - and working with software engineering for an application is difficult and time-consuming.

Software engineering has two parts: software and engineering.

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

Engineering is the development of products using best practices, principles, and methods.

What is software engineering? It is a branch of engineering that deals with the development of software products. It operates within a set of principles, best practices, and methods that have been carefully honed throughout the years, changing as software and technology change.

Software engineering leads to a product that is reliable, efficient, and effective at what it does. While software engineering can lead to products that do not do this, the product will almost always go back into the production stage.

So, what is the complete definition of software engineering?

The IEEE fully defines software engineering as:

1. The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

What the software engineering meaning doesn't explain is that everything that has been software engineered needs to work on real machines in real situations, not within.

Q2. Explain types of software

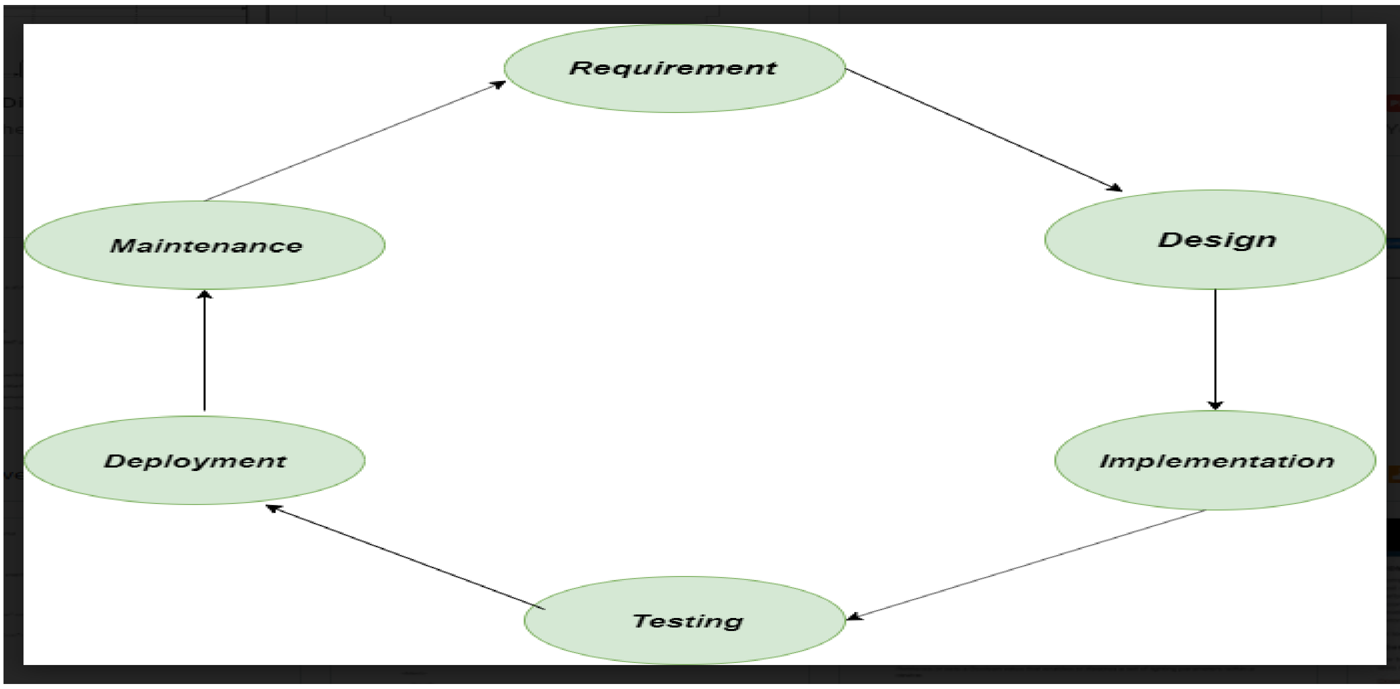
- **Application software.** The most common type of software, application software is a computer software package that performs a specific function for a user, or in some cases, for another application. An application can be self-contained, or it can be a group of programs that run the application for the user. Examples of modern applications include office suites, graphics software, databases and database management programs, web browsers, word processors, software development tools, image editors and communication platforms.

- **System software.** These software programs are designed to run a computer's application programs and hardware. System software coordinates the activities and functions of the hardware and software. In addition, it controls the operations of the computer hardware and provides an environment or platform for all the other types of software to work in. The OS is the best example of system software; it manages all the other computer programs. Other examples of system software include the firmware, computer language translators and system utilities.
- **Driver software.** Also known as device drivers, this software is often considered a type of system software. Device drivers control the devices and peripherals connected to a computer, enabling them to perform their specific tasks. Every device that is connected to a computer needs at least one device driver to function. Examples include software that comes with any nonstandard hardware, including special game controllers, as well as the software that enables standard hardware, such as USB storage devices, keyboards, headphones and printers.
- **Middleware.** The term *middleware* describes software that mediates between application and system software or between two different kinds of application software. For example, middleware enables Microsoft Windows to talk to Excel and Word. It is also used to send a remote work request from an application in a computer that has one kind of OS, to an application in a computer with a different OS. It also enables newer applications to work with legacy ones.
- **Programming software.** Computer programmers use programming software to write code. Programming software and programming tools enable developers to develop, write, test and debug other software programs. Examples of programming software include assemblers, compilers, debuggers and interpreters.

Q3. What is SDLC? Explain each phase of SDLC

Ans. The software development lifecycle (SDLC) is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond. This methodology outlines a series of steps that divide the software development process into tasks you can assign, complete, and measure.

1.To ensure that the software is of high quality: The SDLC includes testing and quality assurance phases, which help to ensure that the software is free of bugs and that it meets the requirements.
2.To manage risks and costs: The SDLC helps organizations to identify and manage risks early in the development process, which can help to reduce costs and minimize the impact of any issues that do arise.
3.To improve communication and collaboration: The SDLC helps to ensure that all stakeholders, including customers, end-users, and developers, are involved in the development process and that their needs are taken into account.
4.To improve efficiency and productivity: The SDLC helps organizations to optimize the use of resources and to streamline the development process, which can improve efficiency and productivity.
5.To increase the likelihood of a successful project outcome: Following a well-defined SDLC process can greatly increase the chances of success of the project, as the process guides the team towards the goal in a systematic and efficient way.
Overall, the **SDLC is a valuable tool** for organizations to use when developing software applications, as it helps **to ensure that the final product is of high quality, meets the requirements**, and is delivered on time and within budget.
The SDLC typically includes the following phases:



1. Requirements gathering and analysis: This phase involves gathering information about the software requirements from stakeholders, such as customers, end-users, and business analysts.
2. Design: In this phase, the software design is created, which includes the overall architecture of the software, data structures, and interfaces. It has two steps:

- **High-level design (HLD):** It gives the architecture of software products.
- **Low-level design (LLD):** It describes how each and every feature in the product should work and every component.

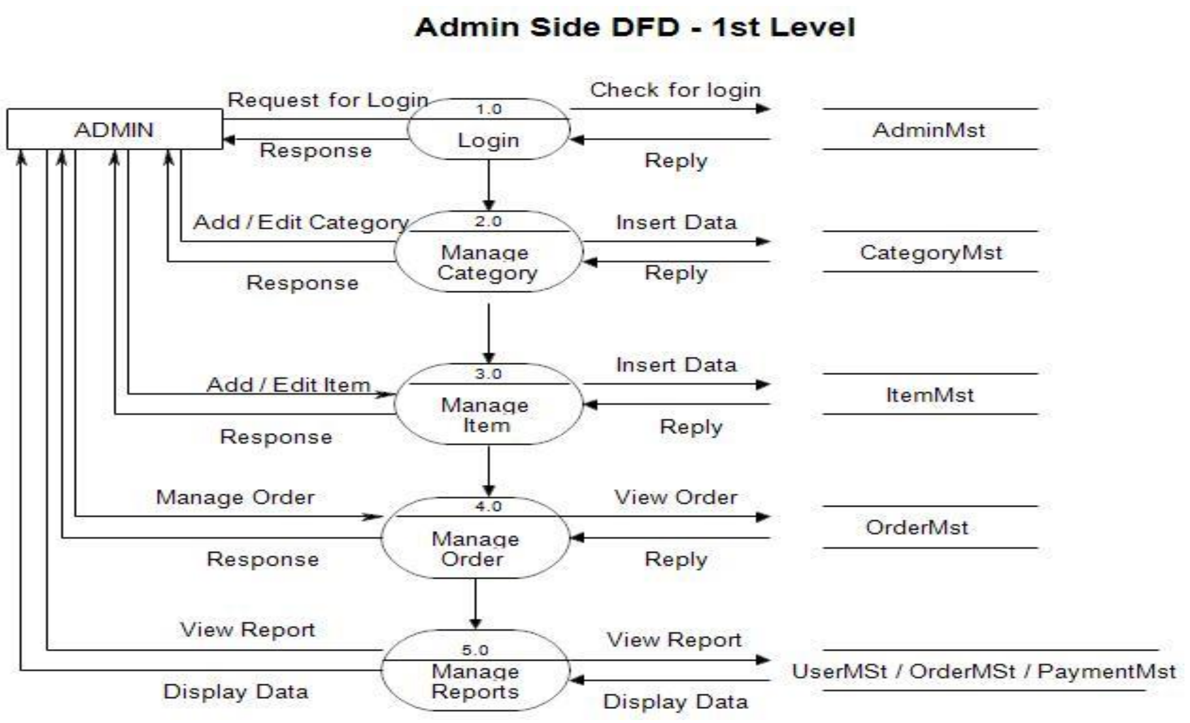
3. Implementation or coding: The design is then implemented in code, usually in several iterations, and this phase is also called as Development. things you need to know about this phase:

- This is the longest phase in SDLC model.
- This phase consists of Front end + Middleware + Back-end.
- **In front-end:** Development of coding is done even SEO settings are done.
- **In Middleware:** They connect both the front end and back end.
- **In the back-end:** A database is created.

4. Testing: The software is thoroughly tested to ensure that it meets the requirements and works correctly.
5. Deployment: After successful testing, The software is deployed to a production environment and made available to end-users.
6. Maintenance: This phase includes ongoing support, bug fixes, and updates to the software.
Q4. What is DFD? Create a DFD diagram on Flipkart
Ans. **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. It is a graphical tool, useful for communicating with users ,managers and other personnel. it is useful for analyzing existing as well as proposed system.
It provides an overview of

- What data is system processes.
- What transformation are performed.
- What data are stored.
- What results are produced , etc.

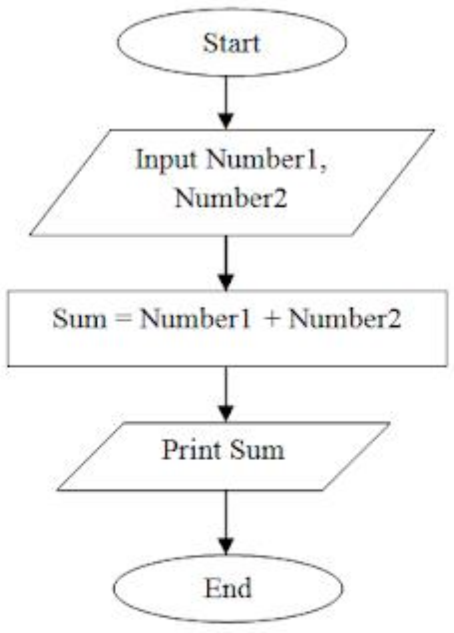
Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modeling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes



Q5. What is Flow chart? Create a flowchart to make addition of two numbers

Ans. *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.*

Flowchart to Add two numbers



Q6. What is Use case Diagram? Create a use-case on bill payment on paytm.

Ans. 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. The actors are often shown as stick figures.