• What is software ? What is software engineering ?

Ans : Software is a set of instructions, data or programs used to operate computers and execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a computer. Software is a generic term used to refer to applications, scripts and programs that run on a device. It can be thought of as the variable part of a computer, while hardware is the invariable part.

Software engineering is defined as a process of analyzing user requirements and then designing, building, and testing software application which will satisfy those requirements. Let’s look at the various definitions of software engineering: IEEE, in its standard 610.12-1990, defines software engineering as the application of a systematic, disciplined, which is a computable approach for the development, operation, and maintenance of software.

Fritz Bauer defined it as ‘the establishment and used standard engineering principles. It helps you to obtain, economically, software which is reliable and works efficiently on the real machines’.

Boehm defines software engineering, which involves, ‘the practical application of scientific knowledge to the creative design and building of computer programs. It also includes associated documentation needed for developing, operating, and maintaining them.

- **Explain types of software**

Ans : Now that we’ve covered the basic definition of software, let’s look at the four main types of software and how they are used in life and in business.

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.

Driver Software :

This software is often considered to be a type of system software. Driver software operates and controls devices that are plugged into a computer. These drivers make it possible for devices to perform their necessary functions. A very good (and practical) example of this is your printer. When you are first setting up your printer to work with your computer, you have to install software to connect the two so that they communicate and print anything you need.

- **what is SDLC ? Explain each phase of SDLC**

Ans : The Software Development Life Cycle (SDLC) refers to a methodology with clearly defined processes for creating high-quality software. in detail, the SDLC methodology focuses on the following phases of software development:

Requirement analysis
Planning
Software design such as architectural design
Software development
Testing
Deployment
This article will explain how SDLC works, dive deeper in each of the phases, and provide you with examples to get a better understanding of each phase.

SDLC Phases

Given below are the various phases:

Requirement gathering and analysis
Design
Implementation or coding
Testing
Deployment
Maintenance

1) <u>Requirement Gathering and Analysis :</u>

During this phase, all the relevant information is collected from the customer to develop a product as per their expectation. Any ambiguities must be resolved in this phase only.
Business analyst and Project Manager set up a meeting with the customer to gather all the information like what the customer wants to build, who will be the end-user, what is the purpose of the product. Before building a product a core understanding or knowledge of the product is very important.

For Example, A customer wants to have an application which involves money transactions. In this case, the requirement has to be clear like what kind of transactions will be done, how it will be done, in which currency it will be done, etc.
Once the requirement gathering is done, an analysis is done to check the feasibility of the development of a product. In case of any ambiguity, a call is set up for further discussion.
Once the requirement is clearly understood, the SRS
(Software Requirement Specification) document is created. This document should be thoroughly understood by the developers and also should be reviewed by the customer for future reference.

2) Design :

In this phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived.

3) Implementation or Coding :

Implementation/Coding starts once the developer gets the Design document. The Software design is translated into source code. All the components of the software are implemented in this phase.

4) Testing :

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

Retesting, regression testing is done until the point at which the software is as per the customer's expectation. Testers refer SRS document to make sure that the software is as per the customer's standard.

5) Deployment :

Once the product is tested, it is deployed in the production environment or first UAT (User Acceptance testing) is done depending on the customer expectation.

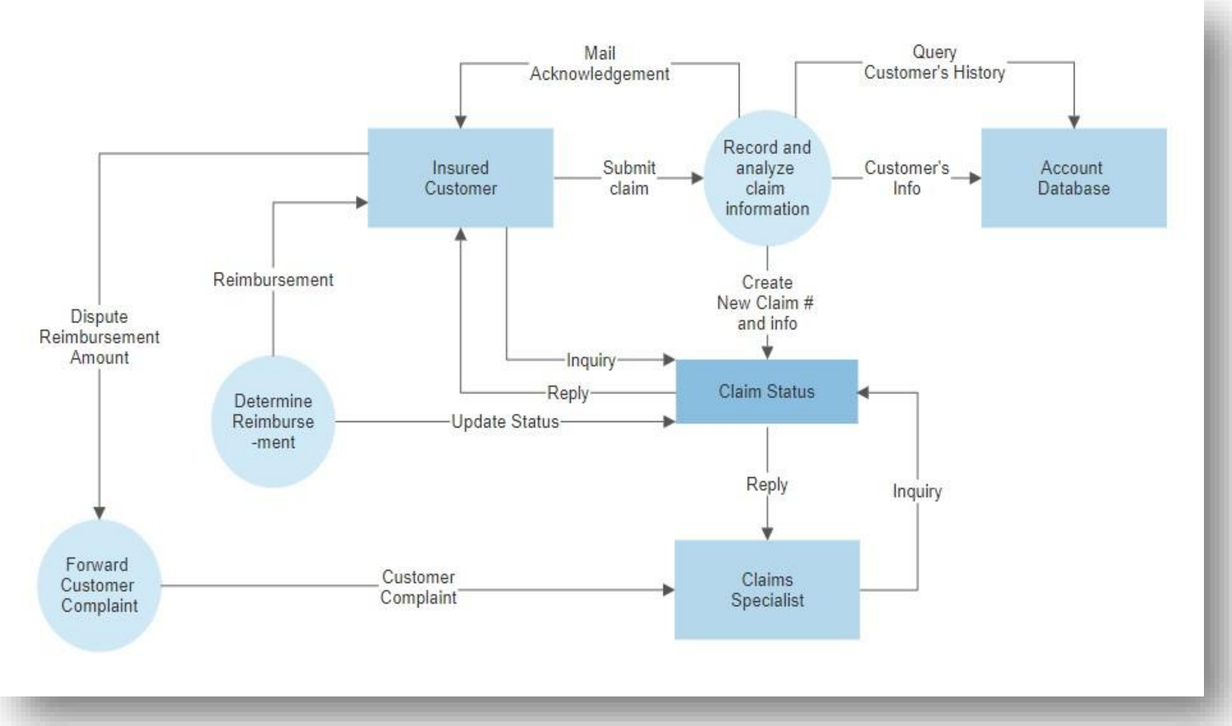
In the case of UAT, a replica of the production environment is created and the customer along with the developers does the testing. If the customer finds the application as expected, then sign off is provided by the customer to go live.

6) Maintenance :

After the deployment of a product on the production environment, maintenance of the product i.e. if any issue comes up and needs to be fixed or any enhancement is to be done is taken care by the developers

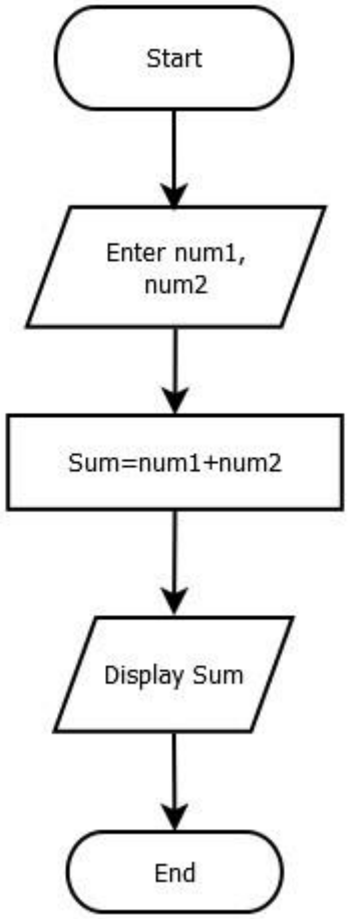
- **what is DFD ? Create a DFD diagram on flipkart.**

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. 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.



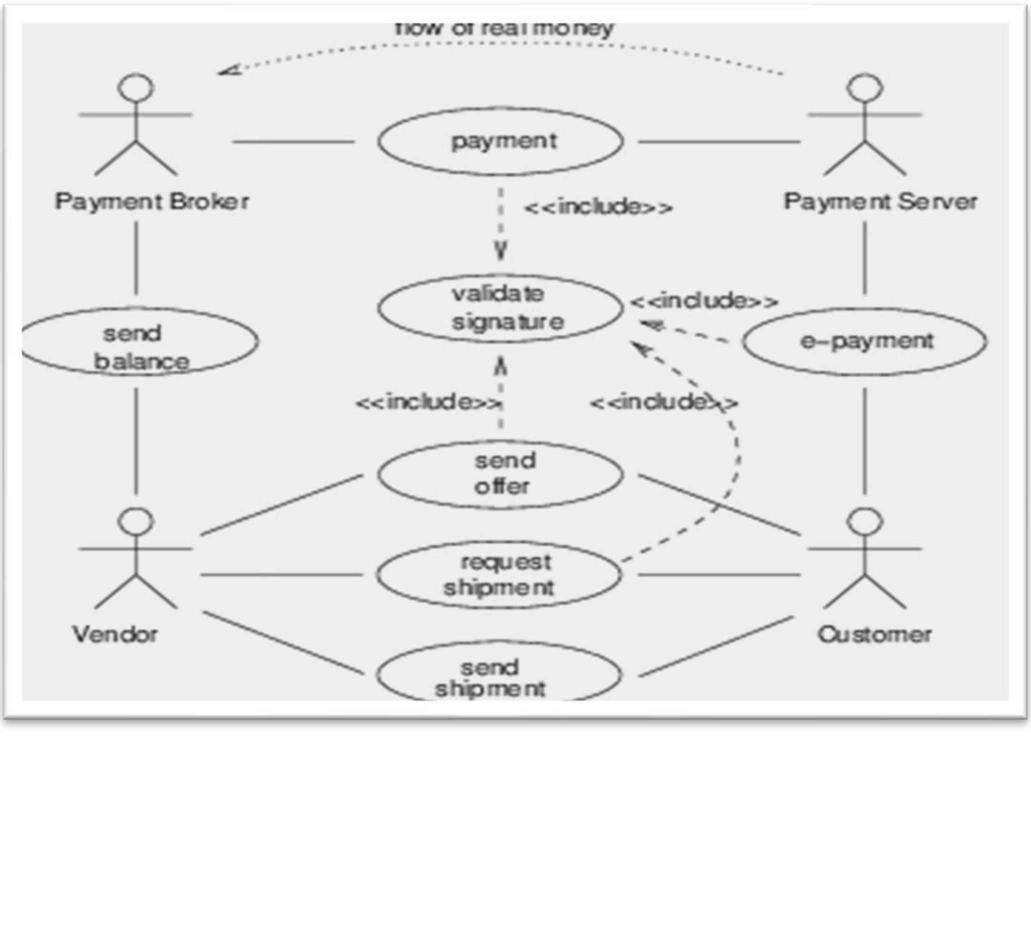
- **what is Flow Chart ? Create a Flowchart to make addition of tow numbers.**

Ans : Flowcharts are nothing but the graphical representation of the data or the algorithm for a better understanding of the code visually. It displays step-by-step solutions to a problem, algorithm, or process. It is a pictorial way of representing steps that are preferred by most beginner-level programmers to understand algorithms of computer science, thus it contributes to troubleshooting the issues in the algorithm. A flowchart is a picture of boxes that indicates the process flow in a sequential manner. Since a flowchart is a pictorial representation of a process or algorithm, it’s easy to interpret and understand the process. To draw a flowchart, certain rules need to be followed which are followed by all professionals to draw a flowchart and is widely accepted all over the countries.



- **what is Use case Diagram ? Creat a Usecase on bill payment on paytm.**

Ans : A use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.


flow of real money
Payment Broker
payment
Payment Server
<<include>>
validate signature
<<include>>
send balance
e–payment
<<include>>
<<include>>
send offer
request shipment
Vendor
Customer
send shipment