**Module-1) Se - Overview Of It Industry**

1. **What is software? What is software engineering?**

**Ans:** **What is software?**

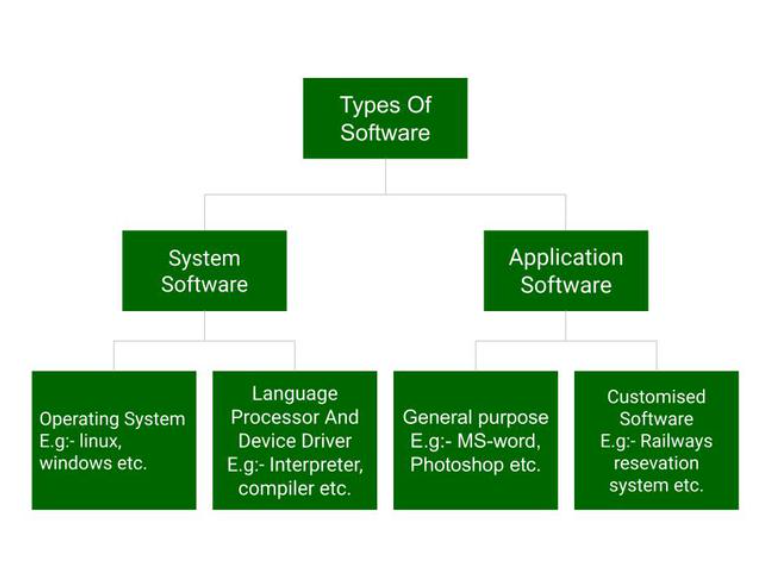
**Software** is a collection of instructions, data, or computer programs that are used to run machines and carry out particular activities. It is the antithesis of hardware, which refers to a computer’s external components. A device’s running programs, scripts, and applications are collectively referred to as “software” in this context.

In a computer science, the software is basically a set of instructions or commands that tell a computer what to do. In other words, the software is a computer program that provides a set of instructions to execute a user’s commands and tell the computer what to do. For example like MS-word MS-Excel, PowerPoint etc.

**What is software Engineering?**

**Software Engineering** is the process of designing, developing, testing, and maintaining software. It is a systematic and disciplined approach to software development that aims to create high-quality, reliable, and maintainable software.

1. Software engineering includes a variety of techniques, tools, and methodologies, including requirements analysis, design, testing, and maintenance.
2. It is a rapidly evolving field, and new tools and technologies are constantly being developed to improve the software development process.
3. By following the principles of software engineering and using the appropriate tools and methodologies, software developers can create high-quality, reliable, and maintainable software that meets the needs of its users.
4. Software Engineering is mainly used for large projects based on software systems rather than single programs or applications.
5. The main goal of Software Engineering is to develop software applications for improving quality, budget, and time efficiency.
6. Software Engineering ensures that the software that has to be built should be consistent, correct, also on budget, on time, and within the required requirements.
7. **Explain types of software**

**Ans:** 

Above is the diagram of types of software. Now we will briefly describe each type and its subtypes:

1. **System Software**
   * Operating System
   * Language Processor
   * Device Driver
2. **Application Software**
   * General Purpose Software
   * Customize Software
   * Utility Software

**System Software**

System Software is software that directly operates the computer Hardware and provides the basic functionality to the users as well as to the other software to operate smoothly. Or in other words, system software basically controls a computer’s internal functioning and also controls hardware devices such as monitors, printers, and storage devices, etc. It is like an interface between hardware and user applications, it helps them to communicate with each other because hardware understands machine language (i.e. 1 or 0) whereas user applications are work in human-readable languages like English, Hindi, German, etc. so system software converts the human-readable language into machine language and vice versa.

**Types of System Software**

It has two subtypes which are:

1. **Operating System:** It is the main program of a computer system. When the computer system ON it is the first software that loads into the computer’s memory. Basically, it manages all the resources such computer memory, CPU, Printer , hard disk, etc., and provides an interface to the user, which helps the user to interact with the computer system. It also provides various services to other computer software. Examples of operating systems are Linux Apple macOS, Microsoft Windows, etc.
2. **Language Processor:**As we know that system software converts the human-readable language into a machine language and vice versa. So, the conversion is done by the language processor. It converts programs written in high-level Programming languages like Java, C, C++ , Python, etc (known as source code), into sets of instructions that are easily readable by machines(known as object code or machine code).
3. **Device Driver:**A Device Driver is a program or software that controls a device and helps that device to perform its functions. Every device like a printer, mouse, modem, etc. needs a driver to connect with the computer system eternally. So, when you connect a new device with your computer system, first you need to install the driver of that device so that your operating system knows how to control or manage that device.

**Features of System Software**

Let us discuss some of the features of System Software:

* System Software is closer to the computer system.
* System Software is written in a low-level language in general.
* System software is difficult to design and understand.
* System software is fast in speed(working speed).
* System software is less interactive for the users in comparison to application software.

**Application Software**

Software that performs special functions or provides functions that are much more than the basic operation of the computer is known as Application Software. Or in other words, application software is designed to perform a specific task for end-users. It is a product or a program that is designed only to fulfill end-users’ requirements. It includes word processors, spreadsheets, database management, inventory, payroll programs, etc.

**Types of Application Software**

There are different types of application software and those are:

1. **General Purpose Software:**This type of application software is used for a variety of tasks and it is not limited to performing a specific task only. For example, MS-Word, MS-Excel, PowerPoint, etc.
2. **Customized Software:**This type of application software is used or designed to perform specific tasks or functions or designed for specific organizations. **Utility Software:**This type of application software is used to support the computer infrastructure. It is designed to analyze, configure, optimize and maintains the system, and take care of its requirements as well.

**Features of Application Software**

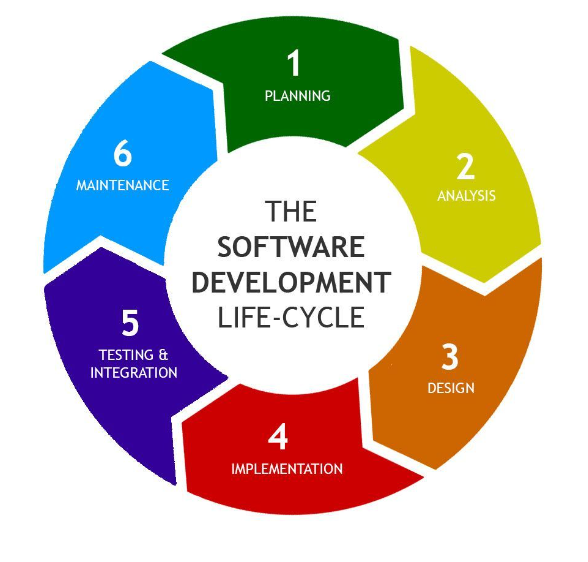
Let us discuss some of the features of Application Software:

* An important feature of application software is it performs more specialized tasks like word processing, spreadsheets, email, etc.
* Mostly, the size of the software is big, so it requires more storage space.
* Application software is more interactive for the users, so it is easy to use and design.
* The application software is easy to design and understand.
* Application software is written in a high-level language in general.

**3.What is SDLC? Explain each phase of SDLC**

**Ans:** **The Software Development Life Cycle (SDLC)**is a process used by software Development organizations to plan, design, develop, test, deploy, and maintain software applications.

**Phases of SDLC:**



**Stage 1. Planning**

The first stage of the SDLC is planning. The purpose of this stage is to develop a basic plan about what an application needs to do based on business requirements.

Development teams formulate high-level plans during this stage. They don't lay out exactly how to implement certain features or functionality or which programming languages are involved.

Instead, teams set basic goals for application features and outline business challenges the features address during this stage. Planning is also a time to identify any unrealistic goals that should be set aside. For instance, consider an application that can't address a business need because of insufficient resources. Development teams should communicate that during this stage, rather than wasting time at a later stage trying to implement a feature they ultimately abandon.

Once development teams have a clear understanding of what the application's purpose is and which features are needed to serve that purpose, the planning stage is complete.

**Stage 2. Analysis**

The analysis stage of the SDLC is where development teams translate high-level plans and goals into actionable ideas. To do this, teams perform a technical analysis of the plans they developed in the previous stage and determine how best to implement them.

The end goal of the analysis step is to have a technical plan that teams can begin putting into action. Line-by-line detail about how to implement code isn't necessary, but teams should know exactly which tools and processes they'll use as they build the app.

**Stage 3. Design**

The design stage focuses on deciding how the application behaves and what it looks like from the user's perspective. For example, if the application has a GUI, teams should sketch what that interface looks like in this stage. Teams should also consider whether users need to register accounts to use the app and, if so, whether there are different types of accounts, such as admin and nonprivileged accounts, for different users.

The design stage draws on technical specifications for the app that teams established during the analysis stage. In fact, some descriptions of the stages of the SDLC treat analysis and design as a single phase. However, because analysis focuses more on technical requirements, while design focuses on UX, developers should separate these processes into two stages.

The design phase is complete when teams have a clear understanding of how users interact with the app.

**Stage 4. Implementation**

The implementation stage -- also sometimes referred to as the *development* or *coding stage* -- is where teams write the actual code. This can be the lengthiest stage of the SDLC if there is a lot of complicated code to write. But it can also be relatively short, especially if teams accelerate code implementation using methodologies such as low-code/no-code or AI-assisted development.

Either way, it's important to establish clear processes for managing code as developers write it during implementation, especially if there are multiple developers working at once. Teams often use a CI server, which merges newly written code into a shared codebase and helps ensure that the code implemented by one developer doesn't cause conflicts with code written by others.

In addition, teams should set clear expectations about coding style and conventions to ensure that all the code is consistent. This also improves developers' ability to maintain or update code that they did not write themselves, which is an important consideration if teams plan to use the application for a period of years during which the original developers may depart the organization.

Implementation is complete once teams have written all the code required to achieve the application's planned functionality. Note, however, that teams may need to come back and adjust some code later, based on the outcomes of the tests performed in the next stage.

**Stage 5. Testing**

After teams have written code, they're ready to test it. Software tests assess how well an application meets goals in areas such as the following:

* **Performance**, which means how responsive the application is.
* **Load**, which refers to the app's ability to perform as demand fluctuates.
* **Security**, which involves detecting security vulnerabilities that may exist in the app.
* **Usability**, which ensures that the application provides acceptable UX.

Teams should design tests that align with the goals established during the first three stages of the SDLC. After they design the tests, teams should run the tests and identify results that don't meet expectations. If an application doesn't pass all tests, teams may need to update some of the code from the previous stage to fix the issue and then run the tests again.

Once the application has successfully passed all tests, the testing phase is complete.

**Stage 6. Deployment**

Deployment is the stage where the application moves into a production environment, where it is accessible to end users.

If the app is hosted using a SaaS model, teams deploy the app into a production hosting environment and then configure network firewalls so that all requests for the app are directed at that environment.

If the app runs locally on users' devices, deployment entails packaging it and distributing the packages to users through channels, such as an app store, where they can search for and download applications.

To mitigate the risk of pushing buggy apps into production, teams should consider methods such as blue/green deployment, which helps to switch over to a new version of an application in a gradual, systematic way, and canary releases, which push out new versions of an application to some users before others, in order to identify bugs before they impact the entire user base.

Containerized applications can also help reduce deployment risks. Containers provide nearly identical hosting environments, no matter where they are deployed. If an application ran in a container successfully during the testing stage, teams can expect it to operate correctly during deployment as well.

Once teams release the application to all target end users, deployment is complete.

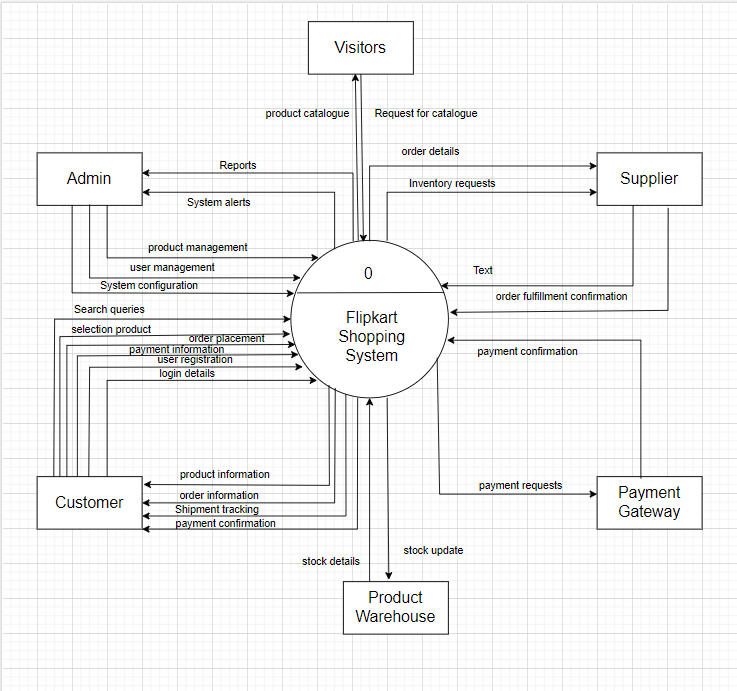
**Stage 7. Maintenance**

The final stage of the SDLC is maintenance. The main goal of maintenance is to monitor the application on a continuous basis to identify issues that arise once it is in production. For example, if a certain type of request triggers an error, development teams should note that so they can fix the issue.

**4. 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.

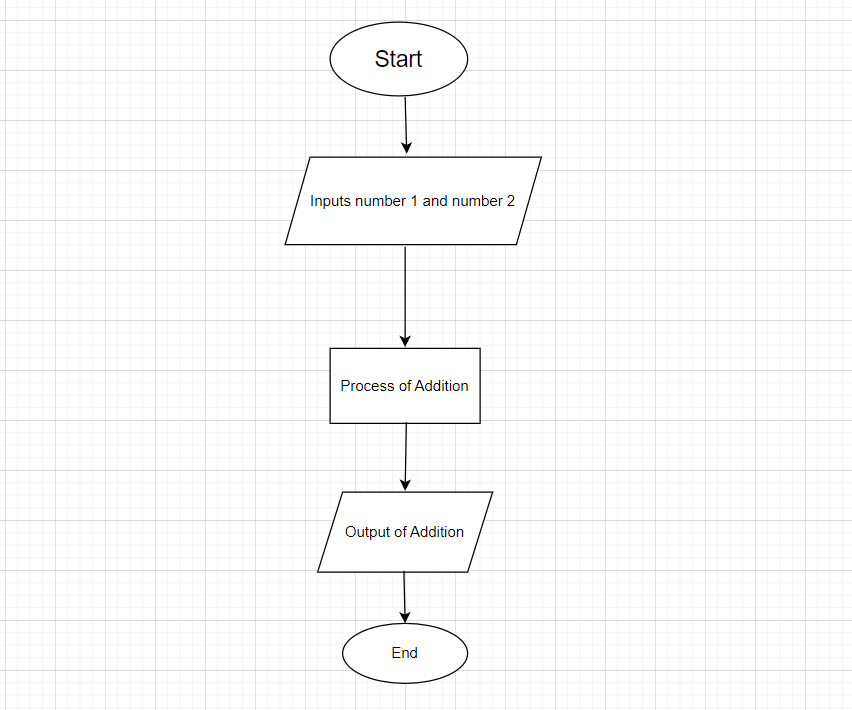
**DFD diagram on Flipkart:**

****

**4.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.

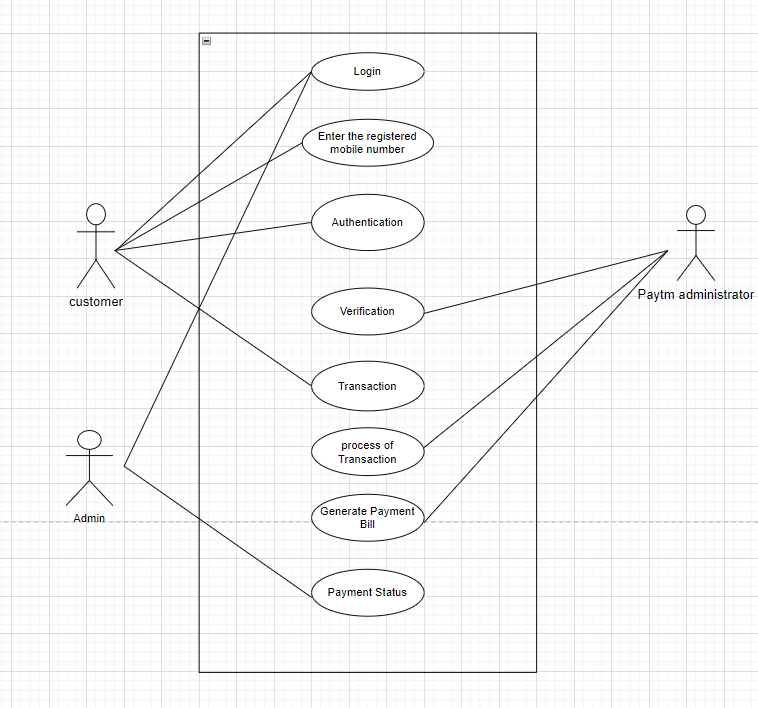
**Flowchart:**

****

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

**Ans:** A use case is a description of how a user interacts with a system or product. Companies build use cases to establish success scenarios, failure scenarios, and any important variants or exceptions.

**Use-case:**

****