1. What is software? What is software engineering?

* **Software**: refers to a set of instructions that tell a computer what to do. It is a collection of programs, data, and instructions that tell a computer how to perform specific tasks.
* **Software engineering:** is the process of designing, creating, testing, and maintaining software. It is a discipline that applies engineering principles to the development of software. The goal of software engineering is to create high-quality software that meets the needs of its users, is reliable, and is easy to maintain.

1. Explain types of software

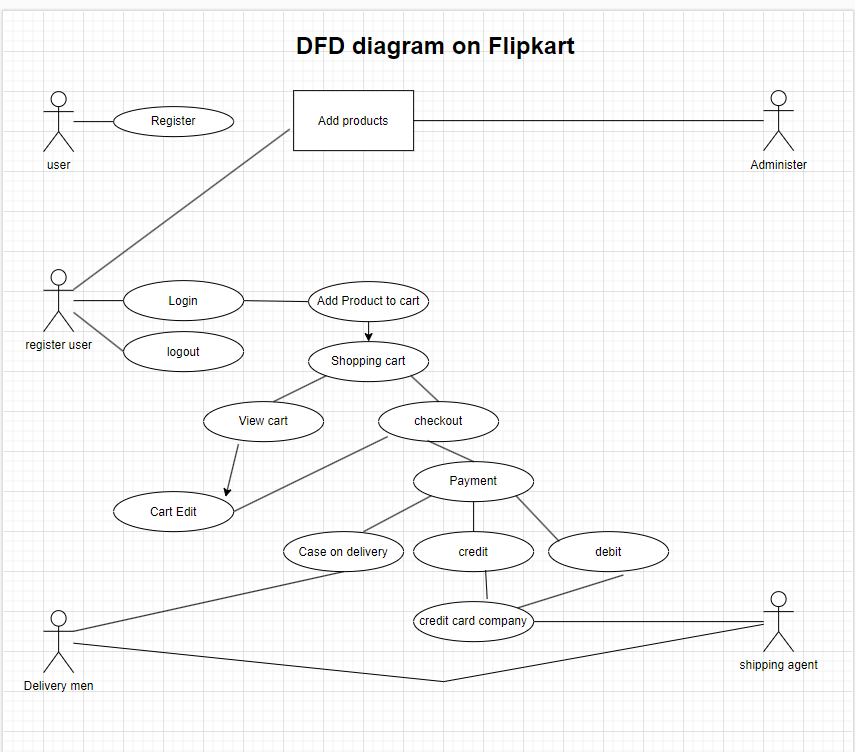
* There are three main types of software: **system software**, **application software**, and **utility software**
* **System software**: is responsible for managing the computer’s resources, such as memory, storage, and processing power. It includes the operating system, which is the primary software that controls the computer’s hardware and provides an interface for the user to interact with the computer. Other examples of system software include device drivers and firmware.
* **Application software**: is designed to perform specific tasks for the user, such as word processing, photo editing, or playing games. It is created to meet the needs of a particular user or group of users. Examples of application software include Microsoft Word, Adobe Photoshop, and Minecraft.
* **Utility software**: is designed to help maintain and optimize the computer’s performance. It includes tools such as antivirus software, disk cleaners, and backup software.

1. What is SDLC? Explain each phase of SDLC

* **Software Development Life Cycle (SDLC):** is a process used by software development organizations to plan, design, develop, test, deploy, and maintain software applications. It consists of seven phases.
* **Planning and requirement analysis**: In this phase, the project team identifies the project’s goals, requirements, and constraints. The team also determines the feasibility of the project and creates a project plan.
* **Design**: In this phase, the team creates a detailed design of the software. The design includes the software architecture, user interface, database design, and other technical specifications.
* **Development**: In this phase, the team writes the code for the software. The code is written according to the design specifications created in the previous phase.
* **Testing**: In this phase, the team tests the software to ensure that it meets the requirements and specifications. The testing includes functional testing, performance testing, and security testing.
* **Deployment**: In this phase, the team deploys the software to the production environment. The deployment includes installation, configuration, and data migration.
* **Maintenance**: In this phase, the team maintains the software by fixing bugs, adding new features, and updating the software to meet changing requirements.
* **Retirement**: In this phase, the team retires the software when it is no longer needed. The team may archive the software or dispose of it.

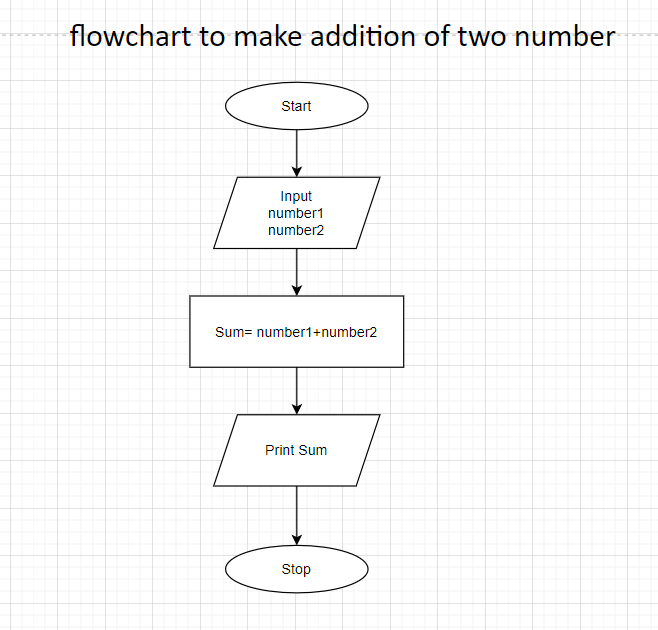
1. What is DFD? Create a DFD diagram on Flipkart

* **Data Flow Diagram (DFD):** is a graphical tool used to represent the flow of data in a system or a process. It provides insight into the inputs, outputs, and transformations of data in a system. DFDs are useful for communicating with users, managers, and other personnel, and for analyzing existing as well as proposed systems.



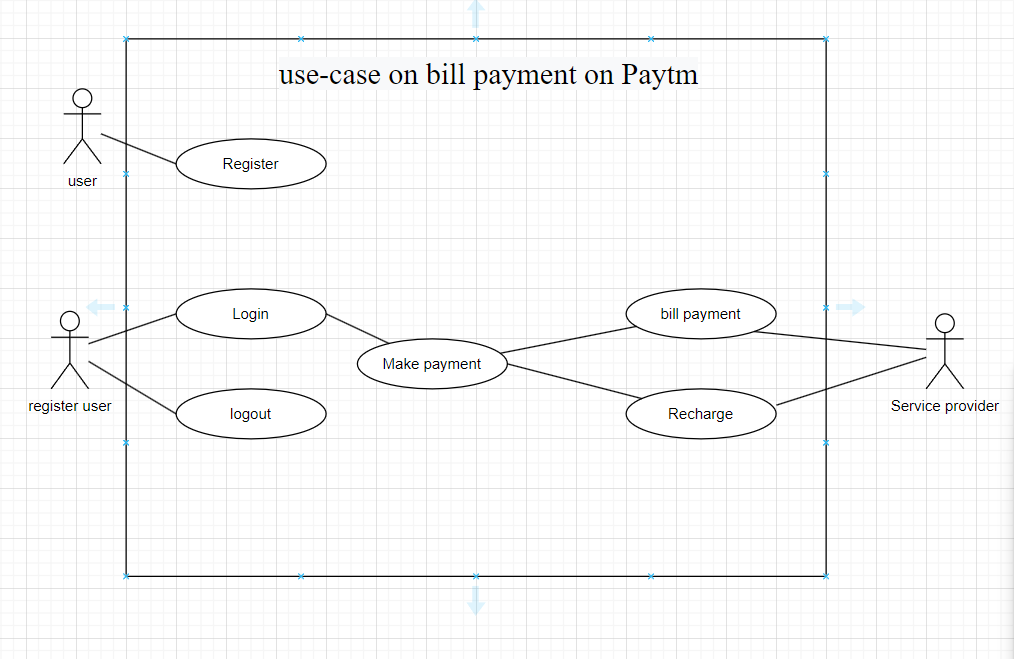
1. What is Flow chart? Create a flowchart to make addition of two numbers

* A flowchart is a visual representation of a process or algorithm. It uses different shapes to represent different types of steps or activities and connects them with arrows to show the flow of the process. Here's a simple flowchart for adding two numbers.



1. What is Use case Diagram? Create a use-case on bill payment on Paytm.

* A use case diagram is a type of Unified Modeling Language (UML) diagram that represents the interactions between users (actors) and a system to achieve specific goals (use cases). It provides a high-level view of a system's functionality and the ways in which users interact with that system.
* **Actor**: An actor is an external entity (such as a user, system, or another software component) that interacts with the system. Actors are represented by stick figures or other symbols, and they participate in one or more use cases.
* **Use Case**: A use case represents a specific functionality or behavior that the system provides to its users. It describes a sequence of actions that the system performs to yield a particular result for a specific actor. Use cases are represented by ovals.
* **Association**: An association line connects an actor to a use case, indicating that the actor participates in that use case.
* **System Boundary**: The system boundary, typically represented by a box or a boundary, encloses the use cases of the system. It defines the scope of the system under consideration.



-----------------------------------END-----------------------------------