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

* **Software is the set of instructions that tell a computer what to do. It includes programs like games, web browsers, and word processors. Without software, computers wouldn't be able to perform tasks or run applications. It's like the brains of the computer, guiding its actions.**
* **Software engineering is like building a house but with computer programs instead of bricks and mortar. It's the process of designing, creating, and maintaining software using specific methods and principles. Just like architects plan and construct buildings, software engineers plan and develop programs, making sure they work well, are easy to use, and can be updated when needed. It involves a lot of problem-solving and creativity to make sure the software meets the needs of its users.**
* **Example :- Think of creating a mobile app like baking a cake: you plan what it should do (like deciding on flavors), then you gather ingredients (like writing code), and finally, you bake it (test and launch). Software engineering is like this process but for apps and programs.**

1. **Explain types of software.**

* **software can be broadly categorized into two main types:**

**1. System Software: This type of software includes operating systems like Windows, macOS, and Linux, which manage the hardware and provide a platform for other software to run on. It also includes device drivers that facilitate communication between hardware devices and the operating system, as well as utility programs like antivirus software, disk defragmenters, and system optimization tools.**

**2. Application Software: Application software refers to programs designed for specific tasks or purposes, such as word processors, web browsers, email clients, spreadsheets, graphics editors, games, and more. These are the programs that users interact with directly to perform various functions on their computers or devices.**

**These two categories cover a wide range of software used in different domains and for various purposes, from managing computer resources to facilitating productivity and entertainment.**

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

* **SDLC stands for Software Development Life Cycle. It's a structured process used by software engineers and developers to design, develop, and maintain high-quality software efficiently. The SDLC consists of several phases, each with its own set of activities and deliverables:**

**1. Requirements Gathering: In this phase, the project team works with stakeholders to gather and document the requirements for the software. This involves understanding the needs of the end-users, defining the scope of the project, and identifying any constraints or limitations.**

**2. Analysis: During this phase, the gathered requirements are analyzed in detail to ensure they are complete, consistent, and feasible. The project team may also conduct feasibility studies to assess the technical, economic, and operational viability of the project.**

**3. Design: In the design phase, the architecture and structure of the software are planned and documented. This includes defining the system architecture, data model, user interface design, and other technical specifications.**

**4. Implementation: This phase involves writing the code according to the design specifications. Developers use programming languages and tools to translate the design into executable code.**

**5. Testing: Once the code is written, it undergoes testing to identify and fix any defects or issues. Testing can include various techniques such as unit testing, integration testing, system testing, and acceptance testing to ensure the software meets the requirements and functions as expected.**

**6. Deployment: After testing, the software is deployed to the production environment. This involves installing the software on the end-users' systems and configuring it for use.**

**7. Maintenance**: The maintenance phase involves maintaining and supporting the software after it has been deployed. This includes fixing bugs, making updates or enhancements, and providing technical support to users.

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

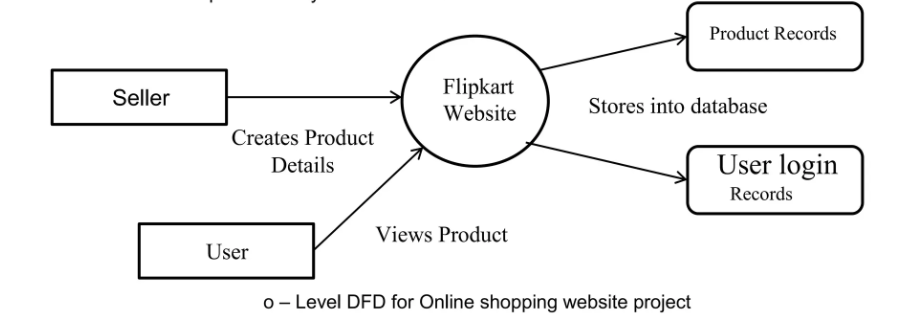
* **DFD stands for Data Flow Diagram. It's a visual representation that illustrates how data flows through a system or process. In a DFD, data is represented as inputs, outputs, data stores, and processes that transform the data.**

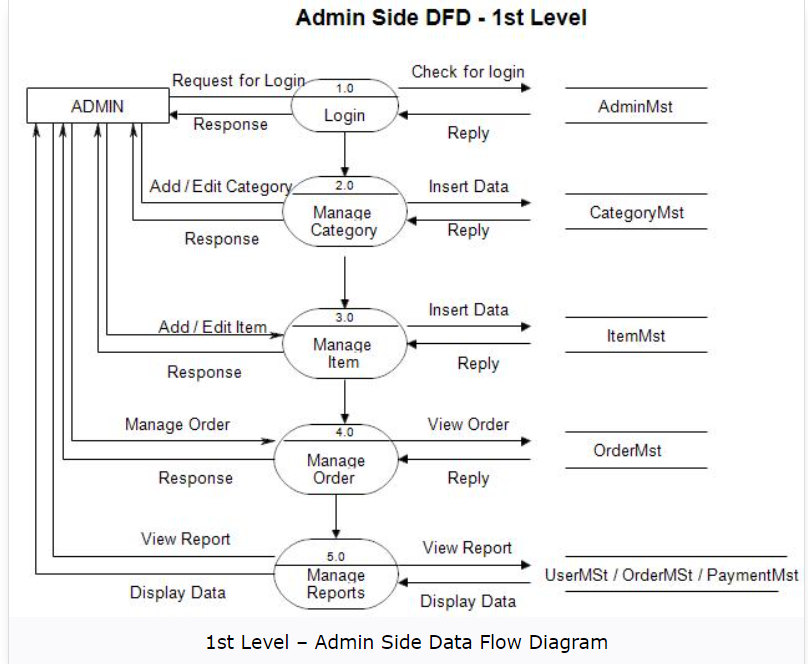
**- Processes: These are represented by rectangles and depict the actions or transformations that occur within the system. Processes can include calculations, data manipulation, or other operations.**

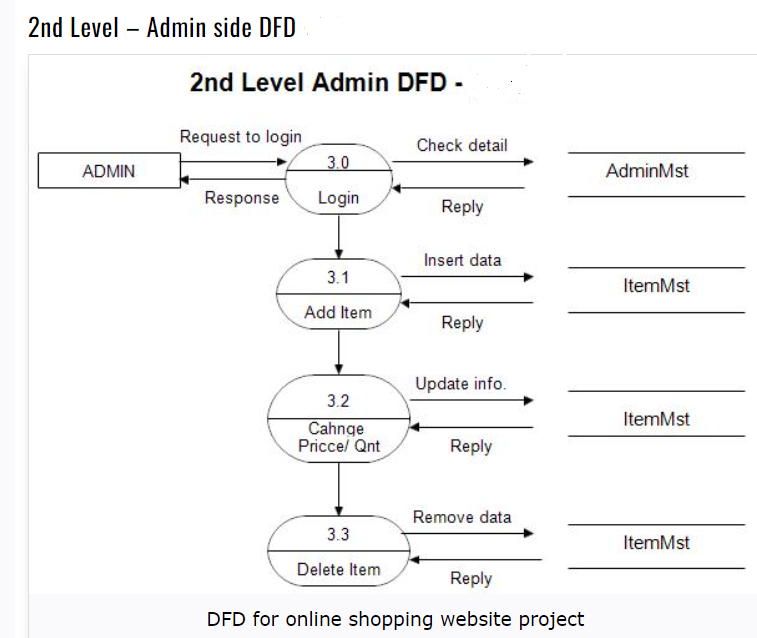
**- Data Flows: These are represented by arrows and show the flow of data between various components of the system, such as inputs, outputs, processes, and data stores.**

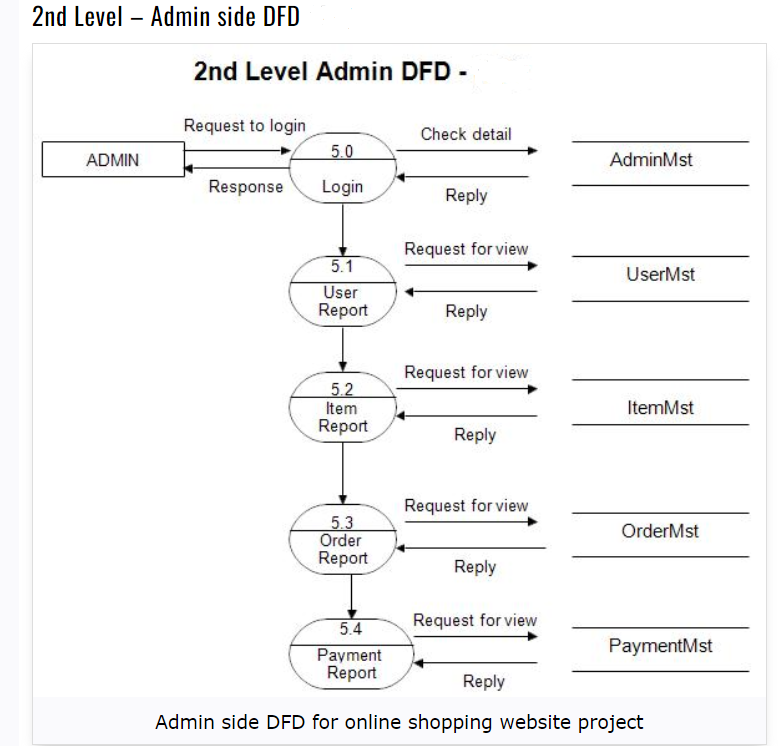
**- Data Stores: These are represented by rectangles with two parallel lines at the top and bottom and represent where data is stored within the system. Data stores can include databases, files, or other repositories.**

**- External Entities: These are represented by squares and represent the sources or destinations of data outside the system. External entities can include users, other systems, or external data sources.**

****

****

****

****

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

**->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.**

**-> Algorithm: A set of finite rules or introduction to followed in calculations or other problem-solving operations"**

**#include<stdio.h>**

**int main()**

**{**

**int n1,n2,ans;**

**printf("Enter 2 value");**

**scanf("%d%d",&n1,&n2);**

**ans=n1+n2;**

**printf("%d",ans);**

**return 0;**

**}**

**algorithm:**

**1)start**

**2)declare**

**3)display enter 2 values**

**4)read value of n1,n2, from user**

**5)ans->n1+n2;**

**6)display ans**

**7)stop**

* **FowChart**



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

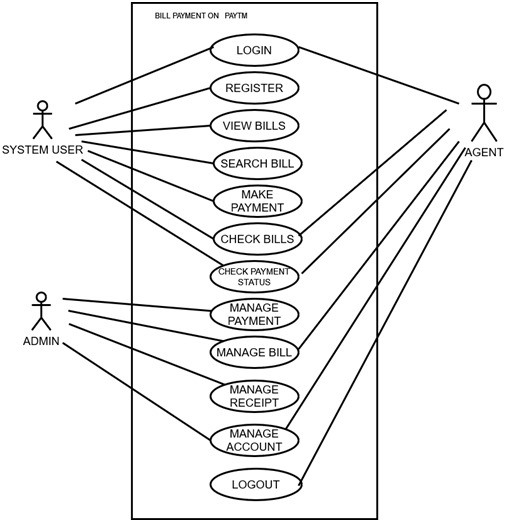
* **A Use Case Diagram is a visual representation that illustrates how users interact with a system or software application to achieve specific goals or tasks. It's a type of diagram used in the Unified Modeling Language (UML), which is a standardized way of visually representing software systems.**
* **In a Use Case Diagram:**

**- Actors: These are represented as stick figures and represent the different types of users or external systems interacting with the software. Actors could be users, administrators, or other systems.**

**- Use Cases: These are represented as ovals and represent the specific tasks or goals that users want to accomplish with the software. Each use case describes a particular interaction between a user and the system.**

**- Relationships: Relationships between actors and use cases are represented by lines connecting them. For example, an actor might initiate or participate in multiple use cases, and a use case might involve multiple actors.**

**Use Case Diagrams are useful for visualizing the functional requirements of a system and understanding how users will interact with it. They help stakeholders, including developers, designers, and clients, to identify system features, define user requirements, and ensure that the software meets the needs of its users.**

****