ASSIGNMENT – 1

**1) What is Software? What is Software Engineering?**

**ANS :-**

Software refers to a set of instructions, programs, or data used to operate computers and perform specific tasks. It encompasses all the intangible components of a computer system that enable it to function and perform various operations. Software can be categorized into two main types:

**1. System Software:** This type of software manages and controls the basic operations of a computer system. It includes operating systems (e.g., Windows, mac OS, Linux), device drivers, utilities, and other fundamental software necessary for the computer to operate.

**2. Application Software:** Application software refers to programs designed to perform specific tasks or applications for users. This includes word processors, spreadsheets, web browsers, games, multimedia players, and specialized applications for various industries such as graphic design, accounting, and engineering.

* Software can be further classified based on its licensing model (proprietary or open-source), its delivery model (installed on local machines or accessed via the cloud), and its purpose (business, entertainment, educational, etc.). Overall, software plays a crucial role in enabling computers and other digital devices to perform a wide range of tasks efficiently.

**-: Software Engineering :-**

Software engineering is a discipline that involves the systematic application of engineering principles, methods, and tools to develop and maintain software systems. It encompasses the entire software development lifecycle, from initial concept and requirements analysis through design, implementation, testing, deployment, and maintenance.

**Key aspects of software engineering include:**

**1. Requirements Engineering:** Gathering, analyzing, and documenting the requirements of the software system to ensure that it meets the needs of its users and stakeholders.

**2. Software Design:** Creating a blueprint or architectural plan for the software system that outlines its structure, components, interfaces, and interactions.

**3. Implementation:** Writing code based on the design specifications using programming languages and development tools.

**4. Testing:** Verifying and validating the software to ensure that it behaves as expected, meets the requirements, and is free of defects or errors.

**5. Deployment:** Installing and configuring the software system in the target environment, including any necessary hardware, software, and network configurations.

**6. Maintenance:** Making modifications, updates, and enhancements to the software system over time to address changing requirements, fix bugs, and improve performance.

Software engineering also involves managing projects, teams, and resources effectively to ensure that software projects are completed on time, within budget, and to the required quality standards. It emphasizes the use of systematic and disciplined approaches to software development, such as software development methodologies (e.g., Agile, Waterfall, DevOps), version control systems, and automated testing techniques.

Overall, software engineering aims to produce high-quality software that is reliable, scalable, maintainable, and meets the needs of its users and stakeholders.

**2) Explain Types of Software.**

**ANS :-**

Certainly! Software can be categorized into various types based on its purpose, functionality, and usage. Here's a comprehensive explanation of some of the most common types of software:

**1. System Software:**

**- Operating Systems (OS):** Operating systems are the core software that manages computer hardware and provides a platform for other software to run. Examples include Windows, macOS, Linux, iOS, and Android.

**- Device Drivers:** Device drivers are software components that facilitate communication between the operating system and hardware devices such as printers, graphics cards, and network adapters.

**- Utilities:** Utilities are software tools that perform specific tasks to manage and maintain the computer system, such as disk management utilities, antivirus software, backup tools, and system optimization utilities.

**2. Application Software:**

**- Word Processors:** Word processing software allows users to create, edit, and format text documents. Examples include Microsoft Word, Google Docs, and LibreOffice Writer.

**- Spreadsheets:** Spreadsheet software enables users to organize data in tabular form, perform calculations, and create charts and graphs. Examples include Microsoft Excel, Google Sheets, and LibreOffice Calc.

**- Database Management Systems (DBMS):** DBMS software allows users to store, retrieve, and manipulate structured data efficiently. Examples include MySQL, Oracle Database, Microsoft SQL Server, and MongoDB.

**- Web Browsers:** Web browsers are software applications used to access and view websites on the internet. Examples include Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.

**- Graphics and Multimedia Software:** Graphics and multimedia software are used to create and edit images, videos, and animations. Examples include Adobe Photoshop, Adobe Premiere Pro, GIMP, and Blender.

**- Communication Software:** Communication software facilitates communication and collaboration among users, such as email clients, instant messaging applications, and video conferencing software. Examples include Microsoft Outlook, Skype, Slack, and Zoom.

**- Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) Software:** ERP and CRM software help businesses manage various aspects of their operations, including finance, human resources, supply chain, sales, and customer relationships. Examples include SAP ERP, Salesforce, and Microsoft Dynamics 365.

**- Educational Software:** Educational software is designed to support teaching and learning activities in educational institutions and includes tools for interactive learning, simulations, and course management. Examples include Moodle, Khan Academy, and Rosetta Stone.

**- Gaming Software:** Gaming software includes video games and entertainment applications that users play for leisure and recreation. Examples include Fortnite, Minecraft, Call of Duty, and Candy Crush Saga.

These are just some examples of the types of software available, and there are many other specialized categories and applications tailored to specific industries, tasks, and user needs.

**3) What is SDLC? Explain Each Phase of SDLC?**

**ANS :-**

SDLC stands for Software Development Life Cycle. It is a process used by software development teams to design, develop, and maintain high-quality software products. The SDLC consists of several phases, each with its own objectives, activities, and deliverables. Here's an explanation of each phase:

**1. Planning Phase:**

**- Objective:** The planning phase involves defining the scope, goals, and requirements of the software project.

**- Activities:** Activities in this phase include gathering and analyzing requirements, identifying project stakeholders, defining project timelines and milestones, and creating a project plan.

**- Deliverables:** The main deliverables of this phase include a project charter, requirements document, project schedule, and initial project plan.

**2. Analysis Phase:**

**- Objective:** The analysis phase focuses on understanding the needs of the end-users and defining the system's functional and non-functional requirements.

**- Activities:** Activities in this phase include conducting interviews with stakeholders, analyzing existing systems (if any), documenting requirements, and creating use cases and user stories.

**- Deliverables:** The main deliverables of this phase include a detailed requirements specification document, use case diagrams, and a functional requirements document.

**3. Design Phase:**

**- Objective:** The design phase involves creating a detailed blueprint or design for the software system based on the requirements identified in the analysis phase.

**- Activities:** Activities in this phase include architectural design, database design, user interface design, and detailed design of system components and modules.

**- Deliverables:** The main deliverables of this phase include system architecture diagrams, database schema, user interface mockups, and detailed design documents.

**4. Implementation Phase:**

**- Objective:** The implementation phase is where the actual coding and development of the software system take place.

**- Activities:** Activities in this phase include writing code, unit testing, integrating system components, and debugging.

**- Deliverables:** The main deliverables of this phase include source code, executable files, and unit test cases.

**5. Testing Phase:**

**- Objective:** The testing phase involves verifying and validating the software to ensure that it meets the specified requirements and is free of defects.

**- Activities:** Activities in this phase include various types of testing such as unit testing, integration testing, system testing, acceptance testing, and regression testing.

**- Deliverables:** The main deliverables of this phase include test plans, test cases, test results, and defect reports.

**6.Deployment Phase:**

**- Objective:** The deployment phase involves installing the software system in the production environment and making it available to end-users.

**- Activities:** Activities in this phase include system deployment, data migration (if applicable), user training, and documentation.

**- Deliverables:** The main deliverables of this phase include a deployed and operational software system, user manuals, and training materials.

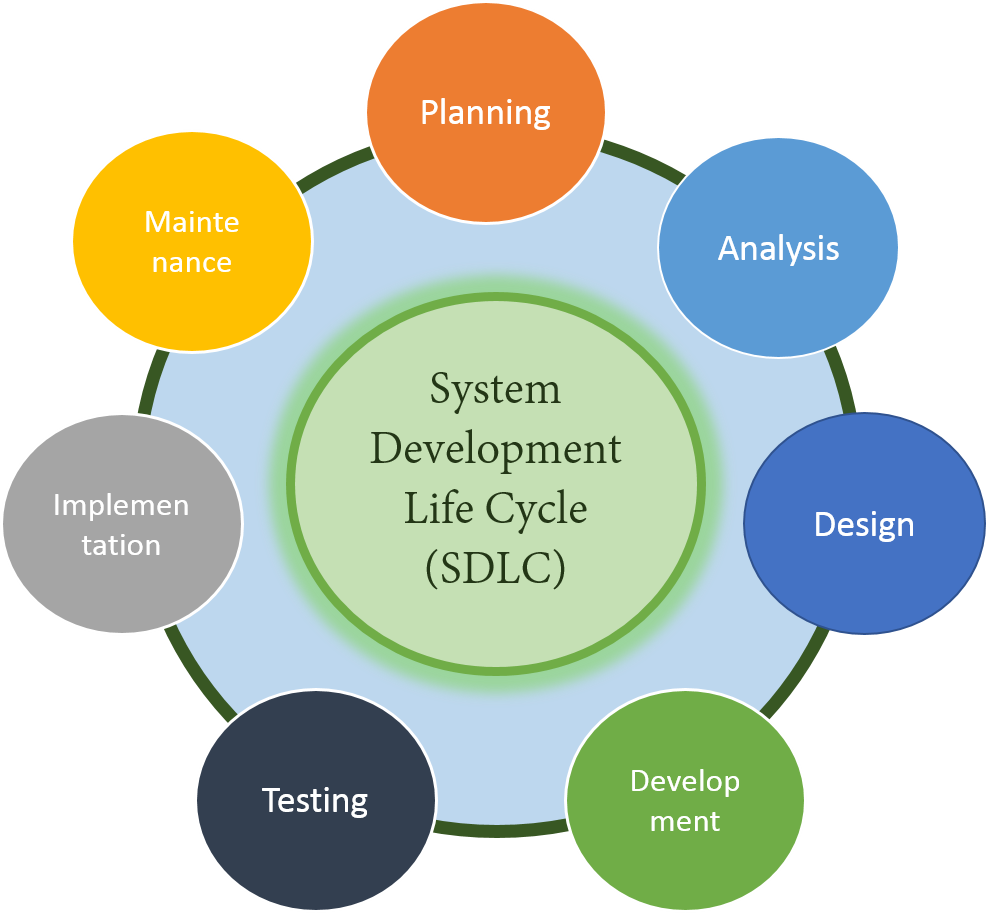
**7. Maintenance Phase:**

**- Objective:** The maintenance phase involves maintaining and supporting the software system after it has been deployed.

**- Activities:** Activities in this phase include fixing defects, implementing enhancements and updates, and providing technical support to users.

**- Deliverables:** The main deliverables of this phase include software patches, updates, and documentation updates.

The SDLC is a cyclical process, meaning that after the maintenance phase, the software may undergo further updates or enhancements, starting the cycle again with the planning phase. Each phase of the SDLC is essential for ensuring the successful development and delivery of high-quality software products.



**5) What is Flow Chart?** **Create a flowchart to make addition of two numbers.**

**ANS :-**

A flowchart is a visual representation of a process or algorithm using various symbols and connecting lines. It is a graphical tool used to depict the steps, decisions, and flow of logic within a system, procedure, or workflow. Flowcharts are widely used in various fields, including software development, engineering, business process management, and education, to help visualize and understand complex processes.

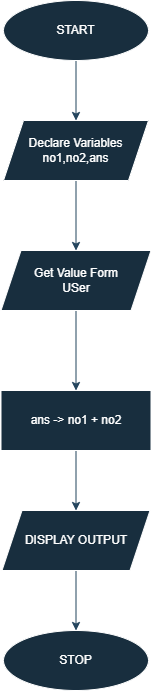
* **Algorithm of flowchart :-**

1. **Start**
2. **Declare Variable num1, num2,Ans**
3. **Get Value Form User**
4. **Display sum1+sum2**
5. **Display Result**

|  |  |  |
| --- | --- | --- |
| **SHAPE** | **DESCRIPTION** | **DESIGN** |
| **Rectangle** | Represent process or  actions |  |
| **Diamond** | Represent Decision  points |  |
| **Oval** | Indicates Start or end  Points |  |
| **Parallelogram** | Represent inputs or  outputs |  |
| **Circle** | Used as connector |  |
| **Arrow** | Indicates flow directions |  |

* **Flow Chart in Details :-**
* **Rectangle (Process)**:
  + **Description**: Rectangles represent processes or actions that occur within the flowchart. These actions can be physical tasks, computational operations, or any other type of activity.
  + **Example**: "Calculate total sales", "Print document", "Verify user credentials".
* **Diamond (Decision)**:
  + **Description**: Diamonds represent decision points where the flow of the process branches into two or more paths based on a condition or criteria. The decision is typically binary, resulting in either a true or false outcome.
  + **Example**: "Is the customer a premium member?", "Is the product in stock?", "Is the balance greater than zero?".
* **Oval (Start/End)**:
  + **Description**: Ovals indicate the beginning or end of a process flow. They mark the starting point where the process initiates and the ending point where it concludes.
  + **Example**: "Start", "End", "Begin process", "Terminate program".
* **Parallelogram (Input/Output)**:
  + **Description**: Parallelograms represent input or output actions within the flowchart. They indicate where data enters or exits the process, such as user inputs, file uploads, or system outputs.
  + **Example**: "Receive user input", "Display error message", "Read data from file", "Write data to database".
* **Circle (Connector)**:
  + **Description**: Circles are used as connectors to link different parts of the flowchart that are not in sequence but are related. They are often labeled with reference numbers or letters to indicate the connection points.
  + **Example**: "Jump to Step 3", "Refer to Process B", "Return to Start".
* **Arrow (Flow Direction)**:
  + **Description**: Arrows are used to indicate the direction of flow within the flowchart, showing the sequence of actions and decisions. They connect the shapes to illustrate the order in which they occur.
  + **Example**: Arrows connect each shape in the flowchart, showing the progression from one step to another.

**-: FLOWCHART :-**

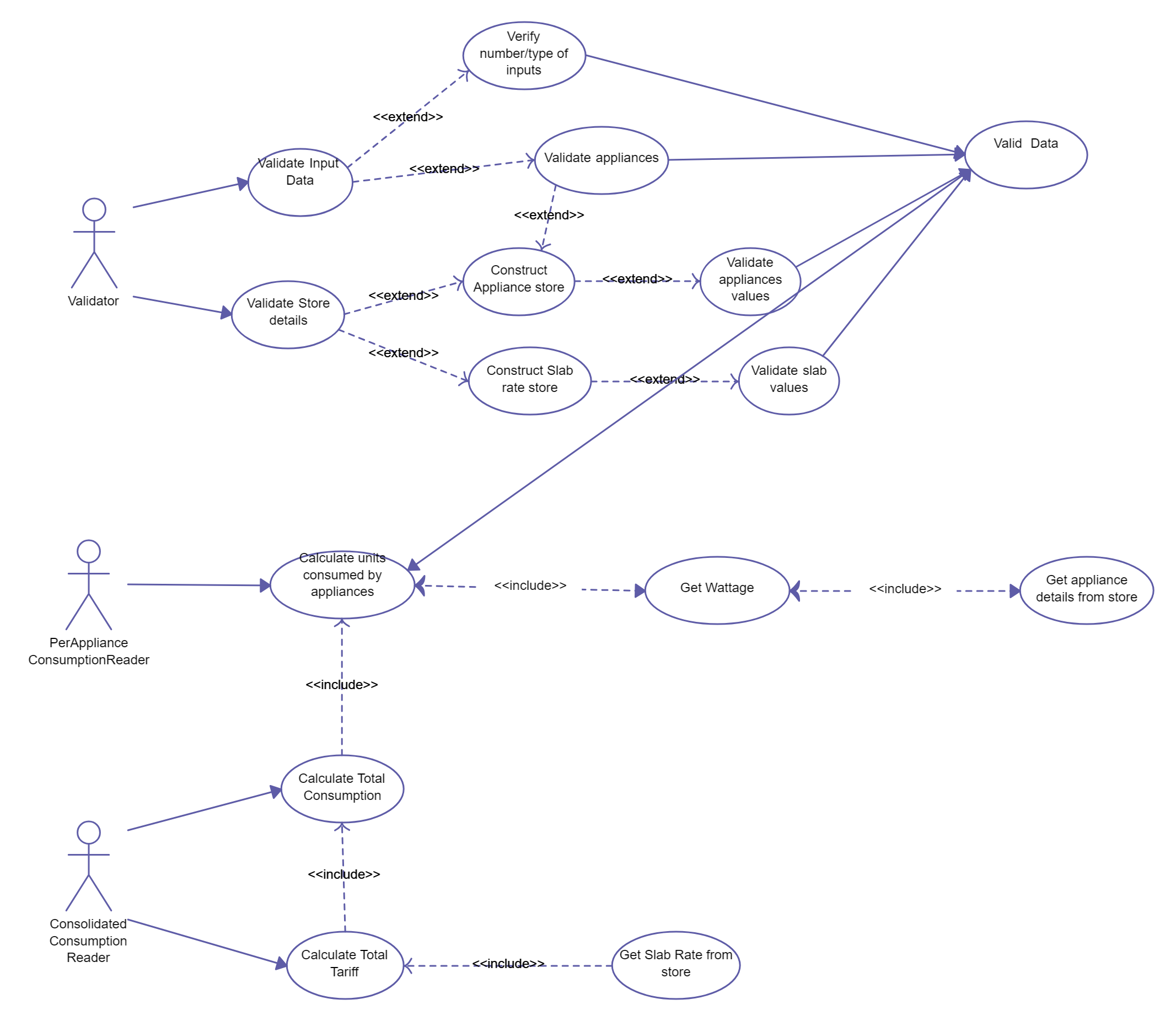


**6) What is USE-CASE diagram? Create a USE-CASE on bill payment on PAYTM.**

**ANS :-**

A Use Case Diagram is a visual representation in Unified Modeling Language (UML) that shows how users (actors) interact with a system to accomplish specific tasks (use cases). It illustrates the relationships between actors and use cases, providing a high-level view of the system's functionality from the user's perspective.

* **The USE-CASE Diagram have shapes like Below:-**
* **Actor**
* **Use case**
* **Relationship**
* **Include Relationship**
* **Extend Relationship**
* **System Boundary**



**4) What is DFD? Create a DFD Diagram on Flipkart.**

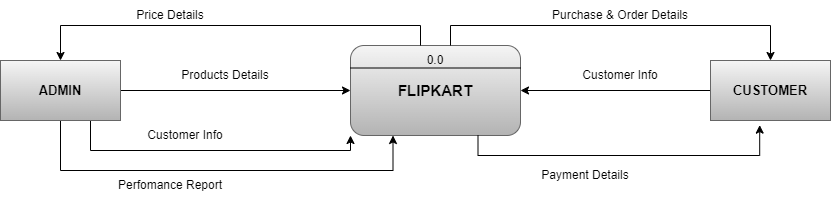
**ANS:-**

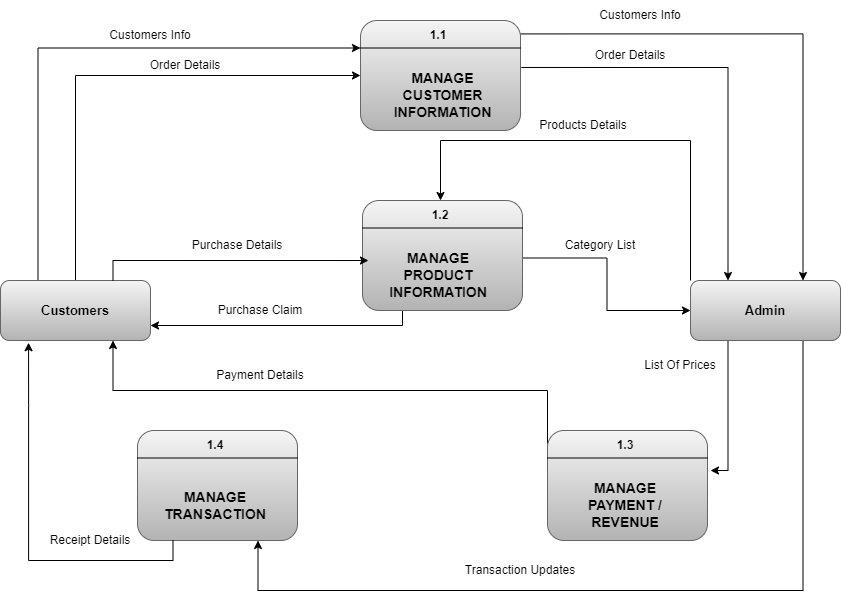
DFD stands for Data Flow Diagram. It's a visual representation of how data flows through a system and how it is processed. In a DFD, data is represented as arrows, processes are represented as squares or circles, and data stores are represented as rectangles. It helps to understand the flow of data within a system, including inputs, outputs, processes, and data storage. DFDs are commonly used in software engineering and system analysis to model and document system architectures and processes.

**There are 4 Types of Data Flow Diagram:-**

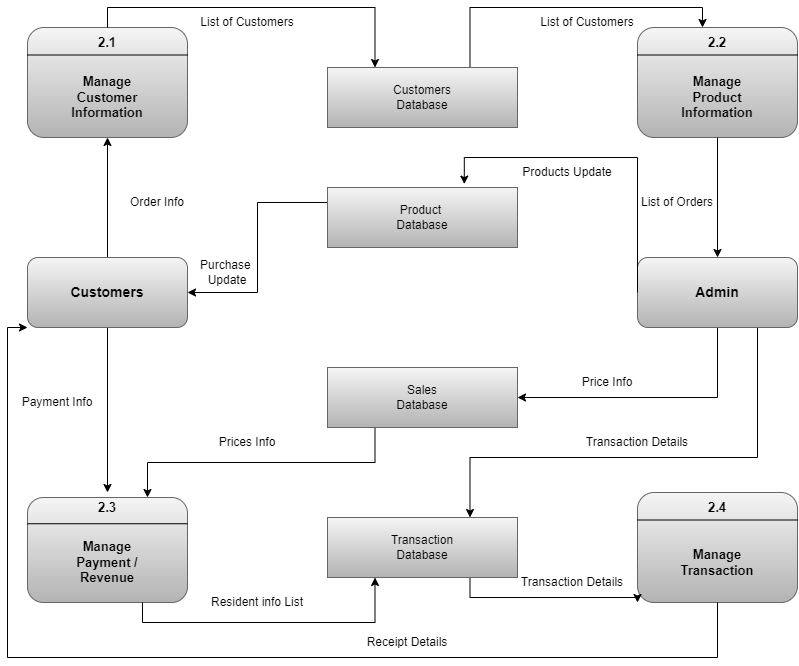
1. **0 LEVEL Data Flow Diagram**
2. **1 LEVEL Data Flow Diagram**
3. **2 LEVEL Data Flow Diagram**

**:- DATA FLOW DIAGRAM LEVEL 0 -:**



**:- DATA FLOW DIAGRAM LEVEL 1 -:**

**:- DATA FLOW DIAGRAM LEVEL 2 -:**

****