MODULE -1

1) What Is Software ? What Is Software engineering ?

Software:

Software refers to a collection of data, programs, and instructions that tell a computer or electronic device how to perform specific tasks. Unlike hardware, which is the physical part of a computer system, software is intangible and exists as code and files that are executed by the hardware.

Software Engineering:

Software engineering is a field of engineering focused on the systematic design, development, testing, and maintenance of software applications and systems. It applies engineering principles to software creation, aiming to produce high-quality, reliable, and efficient software.

~ Key aspects of software engineering include:

1. **Requirements Analysis**: Understanding and documenting what users need from the software. This involves gathering and analyzing requirements to ensure that the final product meets the users' needs and expectations.
2. **Design**: Creating a blueprint for the software system. This includes defining the software architecture, designing user interfaces, and outlining how different components will interact.
3. **Implementation (Coding)**: Writing the actual code based on the design specifications. This involves using programming languages and tools to build the software.
4. **Testing**: Evaluating the software to ensure it works as intended and is free of defects. This includes unit testing (testing individual components),

integration testing (testing interactions between components), and system testing (testing the complete system).

5. **Deployment**: Releasing the software to users. This involves installing and configuring the software in the target environment.
6. **Maintenance**: Updating and fixing the software after it has been deployed. This includes addressing bugs, improving performance, and adding new features as needed.
7. **Project Management**: Planning, scheduling, and managing resources to ensure that the software development project is completed on time, within budget, and to the required quality standards.
8. **Documentation**: Creating detailed records of the software's design, implementation, and usage. This helps with maintenance, future upgrades, and ensures that others can understand and work with the software.

Software engineering emphasizes best practices, methodologies, and tools to ensure that software is developed efficiently and meets high standards of quality. It integrates principles from computer science, project management, and other disciplines to manage the complexities of software development.

2)Explain types of software.

1. System Software

- **Operating Systems**: This is the core software that manages hardware resources and provides a user interface. Examples include Windows, macOS, Linux, and Android.
- **Device Drivers**: These are specialized programs that allow the operating system and software applications to interact with hardware components, such as printers, graphics cards, and storage devices.
- **Utility Programs**: These tools perform system maintenance and optimization tasks, like disk cleanup, file management, and antivirus protection. Examples include Disk Cleanup tools and Malwarebytes.

2. Application Software

- **Productivity Software**: Designed to help users complete tasks efficiently. Examples include word processors (e.g., Microsoft Word), spreadsheets (e.g., Microsoft Excel), and presentation software (e.g., Microsoft PowerPoint).
- **Web Browsers**: Applications used to access and navigate the internet. Examples include Google Chrome, Mozilla Firefox, and Safari.
- **Media Players**: Software used to play audio and video files. Examples include VLC Media Player and Windows Media Player.
- **Games**: Software designed for entertainment. Examples include Fortnite, Minecraft, and The Legend of Zelda.

3. Development Software

- **Integrated Development Environments (IDEs)**: These provide comprehensive facilities for software development, including code editing, debugging, and compilation. Examples include Visual Studio, Eclipse, and IntelliJ IDEA.
- **Compilers**: Tools that translate code written in high-level programming languages into machine code that can be executed by a computer. Examples include GCC (GNU Compiler Collection) and Clang.
- **Version Control Systems**: These tools help manage changes to source code over time, allowing multiple developers to collaborate. Examples include Git (with platforms like GitHub and GitLab) and Subversion (SVN).

4. Middleware

- **Database Management Systems (DBMS)**: Software that manages and organizes data in databases, allowing users to create, read, update, and delete data. Examples include MySQL, PostgreSQL, and Oracle Database.
- **Application Servers**: Middleware that provides an environment for running and managing applications, often in a client-server setup. Examples include Apache Tomcat and IBM WebSphere.
- **Message Brokers**: These facilitate communication between different software applications by handling the routing of messages. Examples include RabbitMQ and Apache Kafka.

5. Utility Software

- **Backup Software**: Tools that create copies of data to prevent loss in case of hardware failure or other issues. Examples include Acronis True Image and EaseUS Todo Backup.
- **System Monitoring Tools**: Software that tracks and reports on system performance and health. Examples include Task Manager (Windows), Activity Monitor (macOS), and Nagios.
- **File Compression Tools**: These reduce the size of files and folders to save storage space or facilitate easier sharing. Examples include WinRAR, 7-Zip, and WinZip.

6. Embedded Software

- **Firmware**: Specialized software embedded into hardware devices to control their functions. Examples include the software in routers, printers, and IoT devices like smart thermostats.

7. Enterprise Software

- **Enterprise Resource Planning (ERP)**: Systems that integrate various business processes and functions into a single unified system. Examples include SAP ERP and Oracle ERP Cloud.
- **Customer Relationship Management (CRM)**: Software designed to manage a company's interactions with current and potential customers. Examples include Salesforce and HubSpot CRM.

8. Educational Software

- **E-Learning Platforms**: Software designed for learning and education, often including courses, quizzes, and interactive content.

Q.3: What is SDLC? Explain each phase of SDLC?

- **SDLC→Software Development Life Cycle.**
- A software life cycle model (also called process model) is a descriptive and diagrammatic representation of the software life cycle.
- A life cycle model represents all the activities required to make a software product.
- Software process models are adjusted to meet the need of software engineers and managers for specific project.
    1. **Planning.**
    2. **Analysis.**
    3. **Designing.**
    4. **Development / Implementation.**
    5. **Testing.**
    6. **Maintenances.**

- ### Planning
- The software development lifecycle It required defining resources, timelines, describing technical and management risks.
- Software Project Plan which defines workflow that is to follow. It describes technical task, risks, resources, product to be produced & work schedule.

- ### Analysis
- The software development lifecycle is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond.

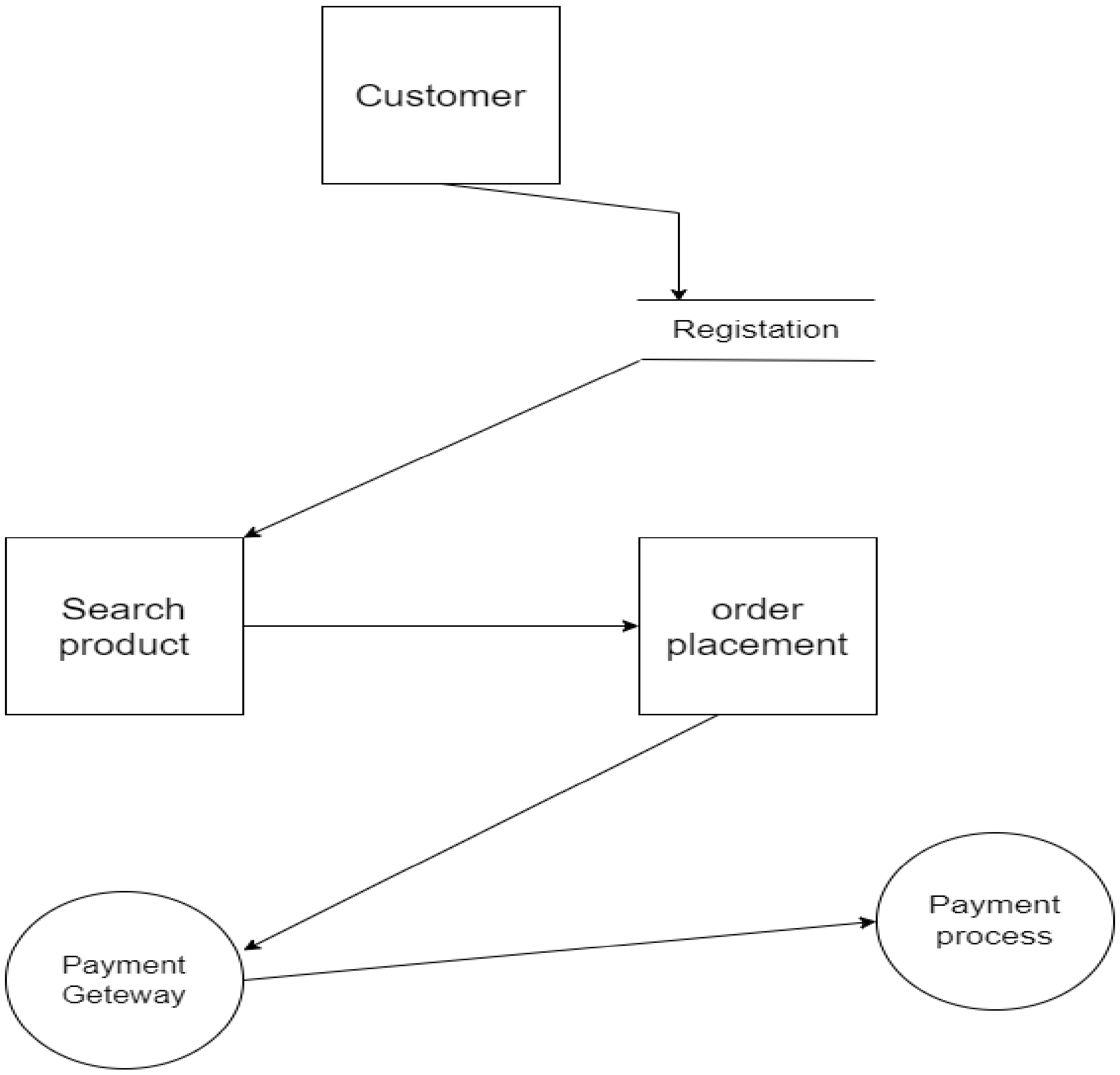
- ### Designing
- The software development lifecycle is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production.

- ### Development / Implementation

- The development phase is where the development team members divide the project into software modules and turn the software requirement into code that makes the product.
- This SDLC phase can take quite a lot of time and specialized development tools. It’s important to have a set timeline and milestones so the software developers understand the expectations and you can keep track of the progress in this stage.
- **Implementation**
- the development team translates the design document into actual code. This process involves writing and configuring software modules, integrating them, and creating a working software system. In the Implementation Phase of SDLC, the development team translates the design document into actual code. This process involves writing and configuring software modules, integrating them, and creating a working software system.
- **Testing**
- The testing phase in a software development life cycle is a crucial stage that helps ensure the software meets the required standards. During this phase, the various components of the software are tested to ensure that they work as expected and conform to the specifications set out in the SDLC. Developers rectify any issues found during this testing phase, and a new version of the software is produced. In this fifth phase of SDLC, the testing is done to ensure that the entire application works according to the customer requirements.
- **Maintenances.**
- One of the primary activities during the maintenance phase is identifying and fixing software bugs or defects that may have been missed during the testing phase or have arisen in the production environment. SDLC occurs after the product is in full operation. Maintenance of software can include software upgrades, repairs, and fixes of the software if it breaks. Software applications often need to be upgraded or integrated with new systems the customer.

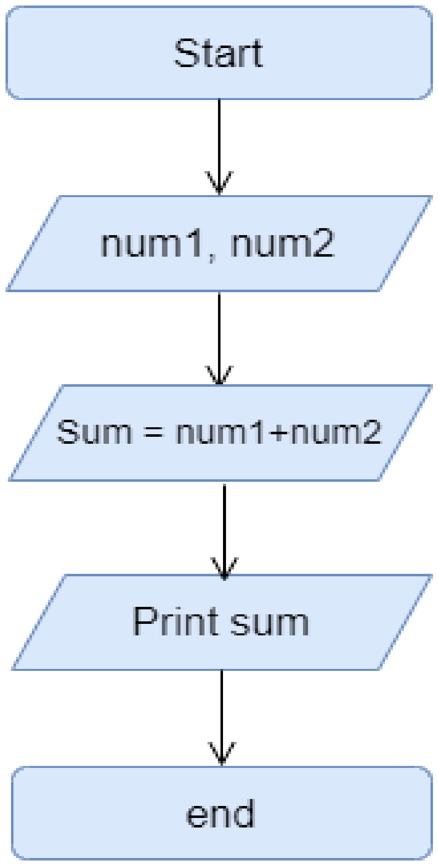
Q.4 What is DFD? Create a DFD diagram on Flipkart?

- **What is DFD?**
- The DFD (also known as a bubble chart) is a hierarchical graphical model of a system that shows the different processing activities or functions that system performs and the data interchange among these functions.
- Each function is considered as a processing station (or process) that consumes some input data and produces some output data. The system is represented in terms of the input data to the system, various processing carried out on these data, and the output data generated by the system.
- **Create a DFD diagram on Flipkart?**



Q.5 What is Flow chart? Create a flowchart to make addition of two numbers?

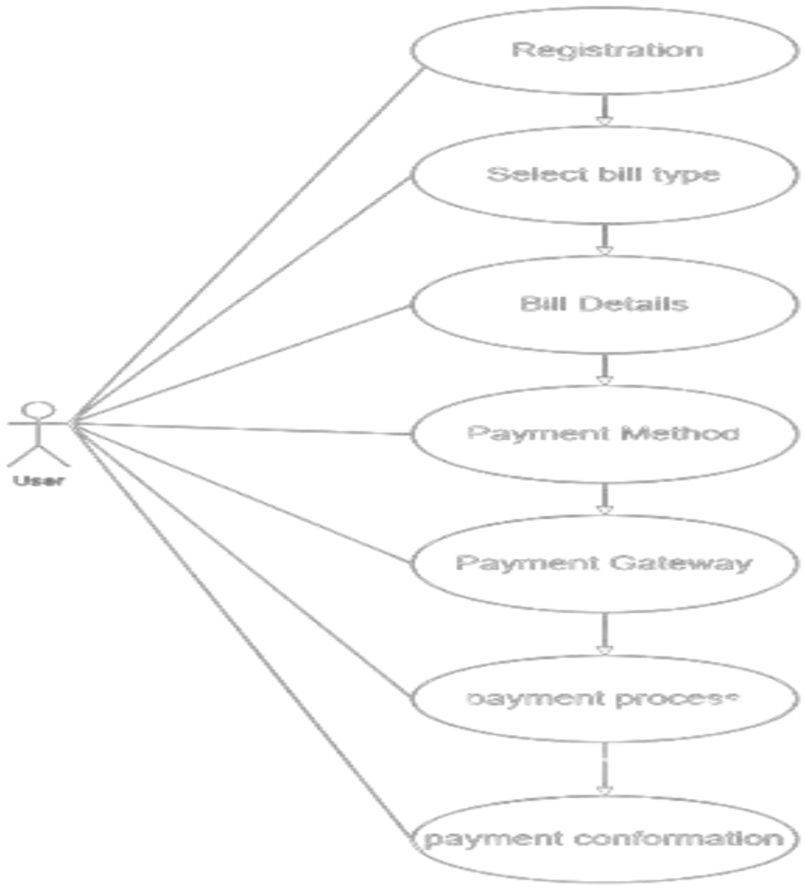
- **What if Flow chart?**
- the flowchart symbols are used to show the steps, order and choices in a process. Together, they form a universal language that makes process analysis easy. I'm sure you've seen flowcharts before with various shapes, lines and arrows to depict stages within a bnprocess like where it begins or ends.
- **Create a flowchart to make addition of two numbers?**



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

- Use case model in UML provides system behaviour.
- Use cases represent the different ways in which a system can be used by the users.
- The purpose of use case is to define the logical behaviour of the system without knowing the internal structure of it.
- A use case represents a sequence of interactions between the user and the system.

Create a use-case on bill payment on Paytm?


Registration
Select bill type
Bill Details
Payment Method
Payment Gateway
payment process
payment conformation
User