**Software Engineering Assignment**

**MODULE: 1**

**SE – Overview of IT Industry**

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

**Answer :** The software is a set of instructions , Data or tasks .

Software engineering is the branch of computer science that deal with the Design , Development , testing and maintenance of software applications .

1. **Explain types of software.**

**Answer :** There are 5 types of software .

* Application software .
* System software .
* Driver software .
* Middleware .
* Programming software .

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

**Answer :** The full form of SDLC is software development lifecycle .These is a structured process that is used to design, develop and test good quality software .

Phases **of SDLC**

1. Planning
2. Analysis
3. Design
4. Implementation
5. Testing & Integration .
6. Maintenance

**Planning** :

The Planning phase is the foundational stage of SDLC where the project is conceptualized and initial resources and strategies are defined.

This involves clearly outlining the goals, deliverables, and boundaries of the project. It helps set expectations and prevents scope creep. Planning involves estimating the necessary budget, time, personnel, and other resources required for the project Developing a detailed project plan that includes timelines, milestones, and tasks is essential for guiding the project throughout its lifecycle.

Assessing the feasibility of the project based on technical, economic, and organizational factors .Allocating resources such as personnel, hardware, and software needed for the project.

**Analysis :**

The Analysis phase follows the Planning phase and focuses on gathering and analyzing requirements from stakeholders.

Thoroughly analyze and understand the functional and non-functional requirements of the software .Determine how the software will interact with users, other systems, and external entities. Rank requirements based on their importance to stakeholders and the overall project goals. Create detailed use cases or user stories to capture how users will interact with the software .We have to Conduct interviews, workshops, and surveys to collect requirements from stakeholders. Analyze and clarify requirements to ensure they are clear, complete, and consistent.

**Design :**

The Design phase involves transforming the requirements specified in the Analysis phase into a blueprint for the software solution. Define the overall structure of the software, including components, modules, and their relationships.**:** Create detailed specifications for each component, defining algorithms, data structures, and interfaces. Design the user interface (UI) to ensure it is intuitive, responsive, and aligns with user expectations. Specify the structure and organization of the database(s) required by the softwars. Create detailed specifications for each component, defining algorithms, data structures, and interfaces. Design the user interface (UI) to ensure it is intuitive, responsive, and aligns with user expect Certainly! Let's explore the phases of Implementation, Testing and Integration, and Maintenance within the Software Development Life Cycle (SDLC):

### 1. Implementation Phase:

The Implementation phase involves the actual coding and development of the software based on the specifications defined in the previous phases. Develop the software according to the detailed design specifications from the Design phase. Write clean, well-documented, and efficient code following coding standards and best practices. Integrate individual modules or components to build the complete system. Prepare the software for testing to ensure it meets the specified requirements. Test individual modules or components to verify their functional Software application or system developed according to design specifications. Test cases and results for individual modules or components. Complete system ready for testing and validation.

### 2. Testing and Integration Phase:

The Testing and Integration phase involves validating the software to ensure it meets quality standards and functions as expected across different levels of testing Detect and identify defects or bugs in the software before it is deployed. Verify that the software meets specified functional and non-functional requirements. Ensure the software is reliable, scalable, and secure through rigorous testing. Test the interactions and interfaces between integrated modules or components.

**Types of Testing:**

* + - **Unit Testing:** Test individual units or modules to ensure they perform as expected.
    - **Integration Testing:** Test interactions between integrated units to ensure they work together seamlessly.
    - **System Testing:** Test the complete system to verify it meets the specified requirements and functions correctly.
    - **Acceptance Testing:** Test the software with end-users to ensure it meets business needs and user expectations.
  + **Test Automation:** Automate testing processes to improve efficiency and repeatability.
  + **Performance Testing:** Test the software's performance under different conditions (load, stress, scalability).
  + **Security Testing:** Test the software for vulnerabilities and ensure it meets security standards.
  + **Test Plans:** Documentation outlining the testing strategy, objectives, and scope.
  + **Test Cases:** Detailed test cases and scenarios to be executed during testing.
  + **Defect Reports:** Reports documenting identified defects or issues and their status.
  + **Test Results:** Results from different testing phases including unit, integration, system, and acceptance tests.

### 3. Maintenance Phase:

The Maintenance phase begins after the software is deployed and involves managing changes, addressing issues, and enhancing the software as needed.Monitor and maintain the software to ensure it continues to function as expected. Address and resolve defects reported by users or identified during monitoring. Make updates and enhancements to the software to meet evolving business needs or technological advancements. Continuously improve performance, scalability, and efficiency of the software. Fixing defects or issues reported by users or identified during operations**.** Making changes to the software to adapt to changes in the environment, hardware, or regulations. Enhancing the software to improve performance, usability, or maintainability based on user feedback. Proactively identifying and addressing potential issues to prevent future problems.

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

ANS->Data Flow Diagram (DFD) represents the flow of data within information systems. Data Flow Diagrams (DFD) provide a graphical representation of the data flow of a system that can be understood by both technical and non-technical users. The models enable software engineers, customers, and users to work together effectively during the analysis and specification of requirements.

## **Types of Data Flow Diagram (DFD)**

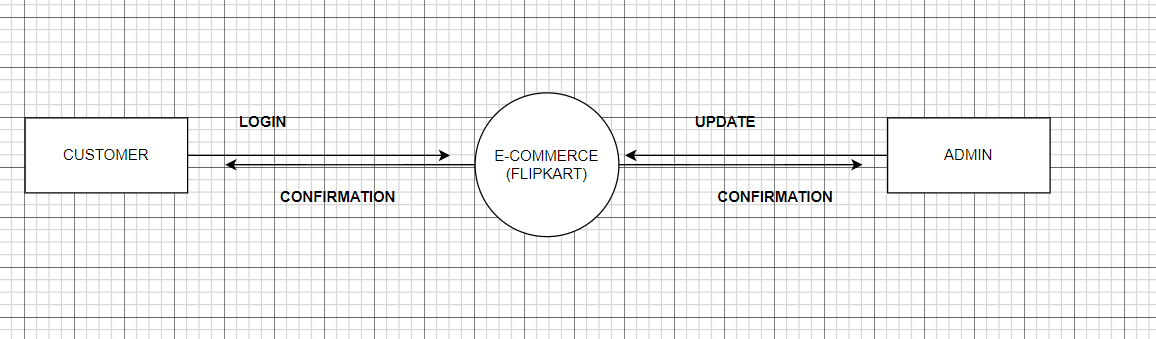
## Components of Data Flow Diagrams (DFD)

The Data Flow Diagram has 4 components

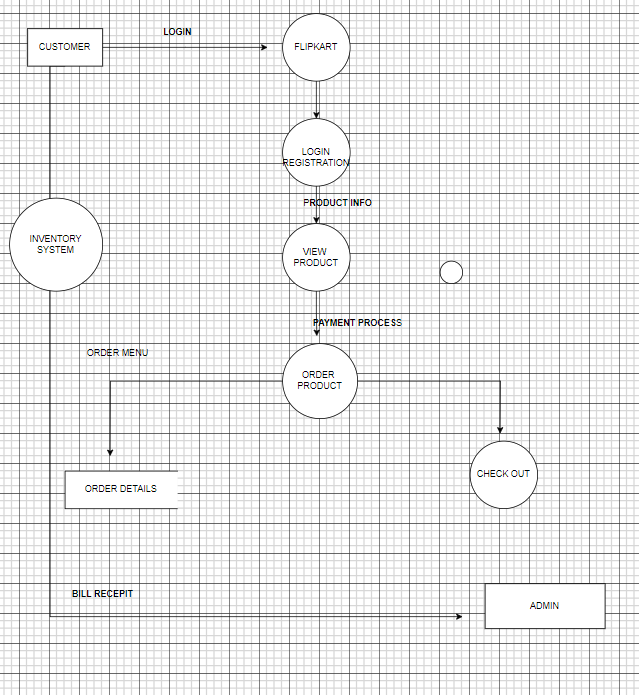
* **Process**
* **Data Flow**
* **Warehouse (Data Store)**
* **Terminator (External Entry):**
* **0 LEVEL DFD**

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

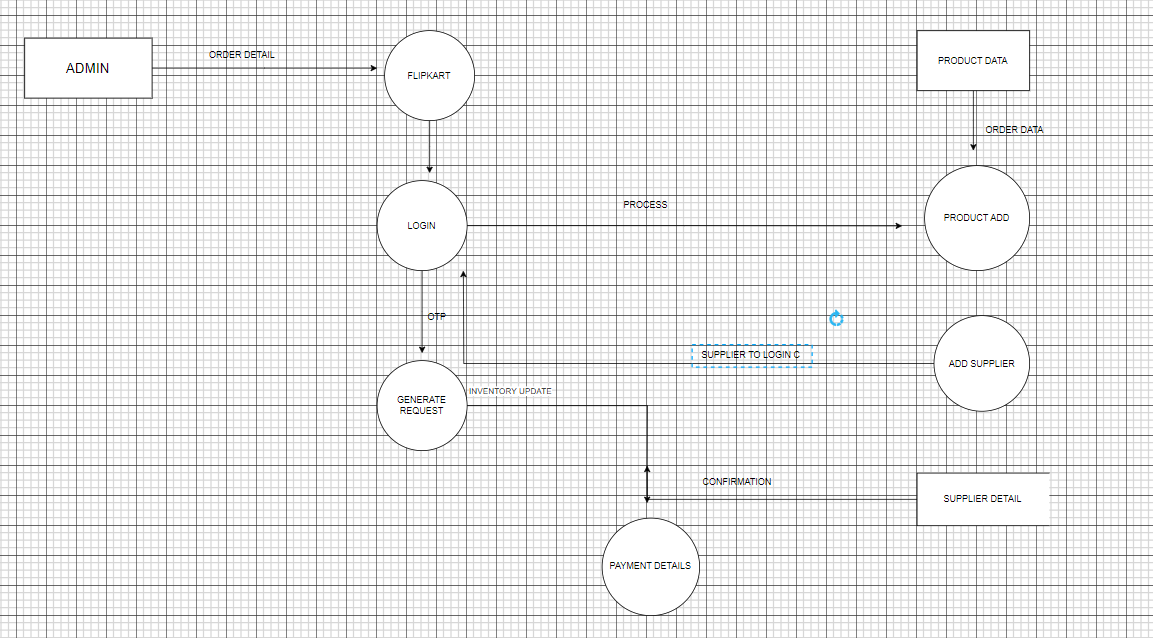
What is Use case Diagram? Create a use-case on bill payment on paytm.



* 1 LEVEL DFD

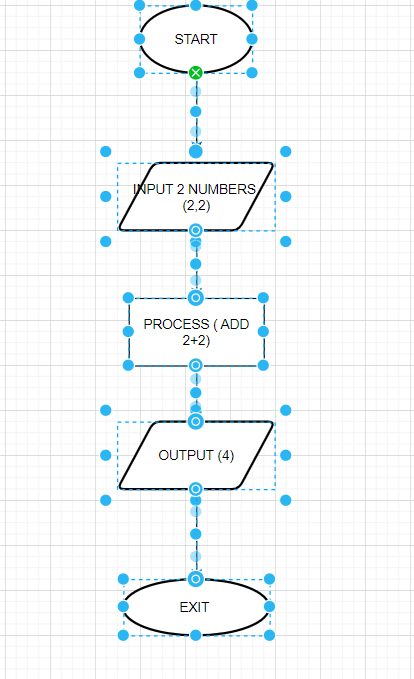
****

* 2 LEVEL DFD



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

A flowchart is a visual representation of a process or algorithm, typically using symbols connected with arrows to show the sequence of steps. It is a diagrammatic representation that helps visualize a process from beginning to end, outlining the decisions, steps, and actions that occur in a clear and concise manner.

Flowcharts typically use standardized symbols for different types of steps and decisions, such as rectangles for actions, diamonds for decisions, circles for starting or ending points, and arrows to indica

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

A use case diagram is a type of behavioral diagram in UML (Unified Modeling Language) that illustrates the interactions between actors (users or external systems) and a system or application. It provides a high-level view of the functionalities and behaviors of the system from the perspective of users.

