SE – Overview of IT Industry

Name – Patel Meet Pankajkumar

Course – Java

Asssignment – 1

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

Ans :-

* Software is like a bunch of computer programs and data that tell a computer what to do, much like how people use language to communicate and give instructions.
* Software engineering is the systematic approach to developing, maintaining, and managing software systems. It involves a structured process that encompasses various phases such as requirements analysis, design, implementation, testing, deployment, and maintenance. Software engineers utilize methodologies, tools, and best practices to ensure that software solutions meet specific requirements, are reliable, scalable, and secure. From understanding user needs to writing code and conducting rigorous testing, software engineering focuses on delivering high-quality software products that add value to users and organizations. It requires a combination of technical expertise, problem-solving skills, and project management capabilities to navigate the complexities of software development and deliver successful outcomes.
* In addition to development, software engineering also encompasses ongoing maintenance and support of software systems. This involves addressing issues, implementing updates, and accommodating changes in requirements or technology throughout the software's lifecycle. Effective project management techniques are essential for overseeing resources, schedules, budgets, and risks to ensure that software projects are delivered on time and within budget. Overall, software engineering plays a vital role in shaping the digital landscape by creating innovative software solutions that drive efficiency, productivity, and growth in various industries and domains.

1. Explain types of software.

Ans :-

* 3 main groups depending on their use and application.

1. System software / operating system.

* Software serves as the essential framework for computer operation, facilitating the translation of inputs from diverse sources into a language comprehensible to machines, while also orchestrating the interaction among various hardware components, exemplified by operating systems like Linux, Windows, macOS, Android, and iOS.

2) Application s/w .

* Software encompasses computer programs designed to fulfill user tasks, including mobile apps for smartphones, desktop applications for standalone use on computers, and web applications accessed via web browsers, such as Google.com and Facebook.com.

3) Programming language.

* Programming involves the sequential steps of designing, coding, testing, debugging, and managing the source code of computer programs, typically written in a programming language, with the goal of crafting software that demonstrates specific intended functionality. Ex. c++, html, java, Simlab, php, Python and Visual basic.

4) Driver Software

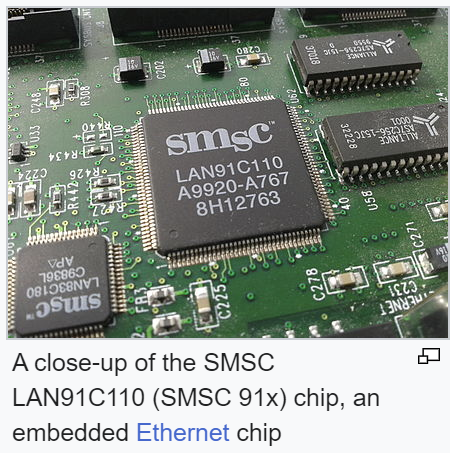
* A **driver** software provides a programming interface to control and manage specific lower-level interfaces that are often linked to a specific type of hardware, or other low-level service. In the case of hardware, the specific subclass of drivers controlling physical or virtual hardware devices are known as device drivers.

5) Middleware

* Middleware is software that lies between an operating system and the applications running on it. Essentially functioning as hidden translation layer, middleware enables communication and data management for distributed applications.

6) Embedded Software

* Embedded software is specialized software that is embedded within hardware devices to control their operation. It is commonly found in consumer electronics, automotive systems, medical devices, industrial machinery, and IoT (Internet of Things) devices.



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

Ans :-

* The Software Development Life Cycle (SDLC) is a structured framework and systematic approach used in software engineering to guide the development process of software applications from inception to retirement. It encompasses a series of distinct phases, each with its own set of activities, deliverables, and objectives, aimed at ensuring the successful delivery of high-quality software products.
* The SDLC serves as a roadmap for software development teams, providing a clear and organized methodology for managing the entire lifecycle of a software project. It enables teams to systematically plan, design, develop, test, deploy, and maintain software systems while adhering to predefined processes and standards.
* It consists of several phases, each with its own set of activities, deliverables, and objectives. Let's elaborate on each phase:
* A diagram of a company

  Description automatically generated

1.Requirements Collection/Gathering (Establish Customer Needs):

* This phase involves gathering and understanding the requirements of the software from stakeholders, users, and customers. It's crucial to establish clear communication channels and techniques such as interviews, surveys, and workshops to elicit requirements effectively.
* The objective is to document the needs, constraints, and expectations of the software system comprehensively and accurately.
* Stakeholder Identification: Identifying all stakeholders involved in the project, including end-users, customers, managers, and developers.
* Requirement Elicitation Techniques: Using various methods such as interviews, surveys, questionnaires, and workshops to gather requirements effectively.
* Requirement Prioritization: Ranking requirements based on their importance and impact on the project's success.
* Requirement Validation: Ensuring that gathered requirements are accurate, complete, and consistent with stakeholders' expectations.
* Functional Requirements Analysis: Defining the specific functions and capabilities that the software must perform.
* Non-Functional Requirements Analysis: Identifying constraints, quality attributes, and performance metrics that the software must adhere to.

2. Analysis (Model And Specify the requirements - "What"):

* In the analysis phase, the collected requirements are analyzed, clarified, and organized into a structured format. This includes identifying the scope of the project, prioritizing requirements, and resolving conflicts or ambiguities.
* The outcome of this phase is a detailed and unambiguous requirements specification document that serves as a blueprint for the subsequent phases of the SDLC.
* Domain Analysis: Understanding the business domain and industry-specific requirements that influence the software design.
* Requirements Specification: Documenting requirements in a clear, unambiguous, and structured format using tools like Use Case diagrams, User Stories, and Requirement Traceability Matrix.

3. Design (Model And Specify a Solution - "Why"):

* Designing phase involves creating a detailed blueprint or plan for how the software will be structured and how its components will interact with each other to fulfill the specified requirements.
* This phase includes architectural design, which defines the overall structure of the software system, as well as detailed design, which specifies the implementation details of individual components.
* The output of this phase is design documents, including diagrams, prototypes, and specifications, which guide the development process.
* Architectural Design: Defining the overall structure of the software system, including components, interfaces, and interactions.
* Detailed Design: Specifying the internal workings of each component, including algorithms, data structures, and module interfaces.
* User Interface Design: Creating intuitive and user-friendly interfaces that enable efficient interaction with the software.
* Database Design: Designing the database schema, tables, relationships, and data access mechanisms required by the software.

4. Implementation (Construct a Solution In Software):

* The implementation phase involves translating the design specifications into executable code using programming languages, frameworks, and development tools.
* Developers write, compile, and integrate code modules according to the design, following coding standards and best practices.
* This phase also involves tasks such as database design, user interface development, and integration with external systems.
* Coding: Writing source code using programming languages such as Java, Python, C++, etc.
* Integration: Combining individual code modules into a unified system and testing their interactions.
* Version Control: Managing changes to the codebase using version control systems like Git, SVN, etc.

5. Testing (Validate the solution against the requirements):

* Testing phase is dedicated to verifying and validating that the software meets the specified requirements and functions correctly.
* Various testing techniques such as unit testing, integration testing, system testing, and acceptance testing are employed to identify defects, errors, and deviations from the expected behavior.
* The goal is to ensure the quality, reliability, and robustness of the software before it is deployed to users.
* Unit Testing: Testing individual units or components of code in isolation to verify their correctness.
* Integration Testing: Testing the interactions between different modules or components to ensure they work together as expected.
* System Testing: Testing the entire software system as a whole to validate its functionality, performance, and reliability.
* User Acceptance Testing (UAT): Involving end-users to validate that the software meets their requirements and expectations.

6. Maintenance (Repair defects and adapt the solution to the new requirement):

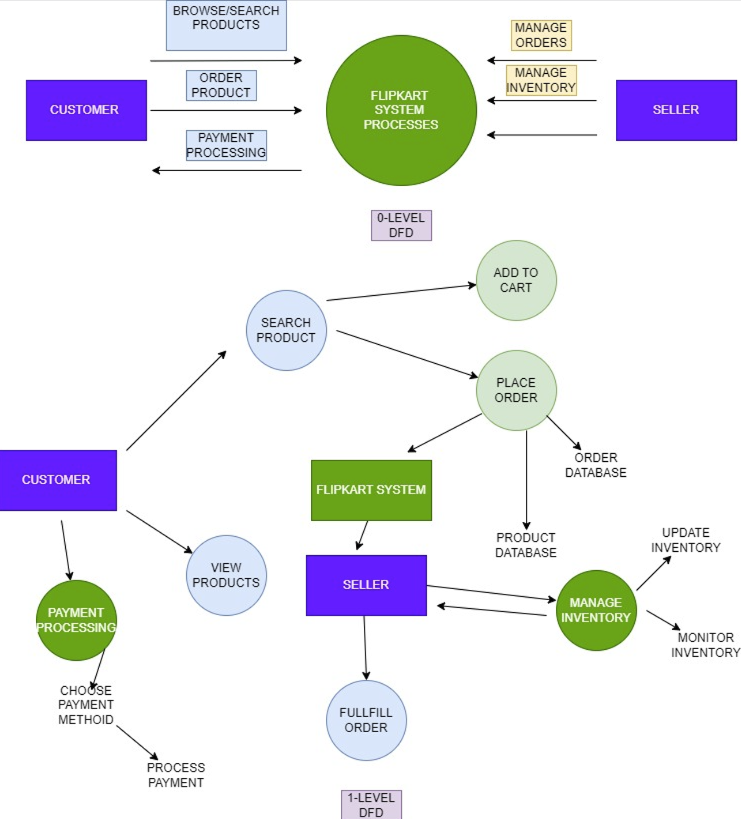
* Maintenance phase involves making modifications to the software after it has been deployed to fix defects, enhance functionality, or adapt to changes in the environment or user needs.
* Activities in this phase include bug fixing, performance optimization, security updates, and feature enhancements.
* The objective is to ensure that the software remains effective, efficient, and aligned with evolving requirements throughout its lifecycle.
* Corrective Maintenance: Fixing defects, bugs, and errors discovered in the software during testing or post-deployment.
* Adaptive Maintenance: Modifying the software to adapt to changes in the operating environment, hardware, or user requirements.
* Perfective Maintenance: Enhancing the software's functionality, performance, and usability to meet evolving needs and improve user satisfaction.
* Preventive Maintenance: Proactively identifying and addressing potential issues or weaknesses in the software to prevent future problems.

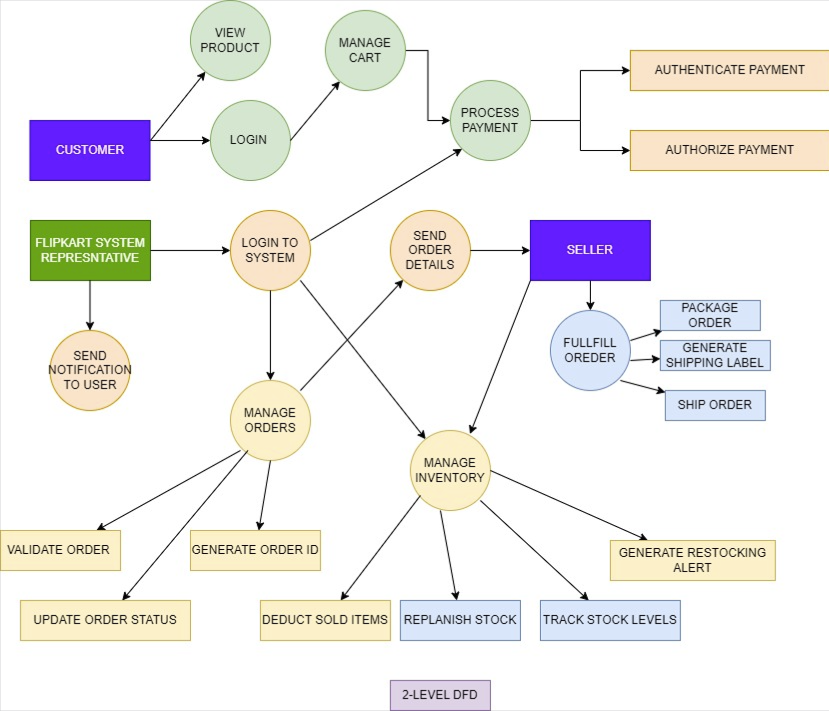
In summary, the SDLC provides a systematic approach to software development, guiding teams through distinct phases of requirements analysis, design, implementation, testing, and maintenance, with each phase contributing to the overall success of the software project.

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

Ans :-

A Data Flow Diagram (DFD) is a graphical representation of the flow of data through a system. It illustrates how data is input into the system, processed, stored, and output. DFDs are commonly used in software engineering to visualize the structure and operation of information systems.

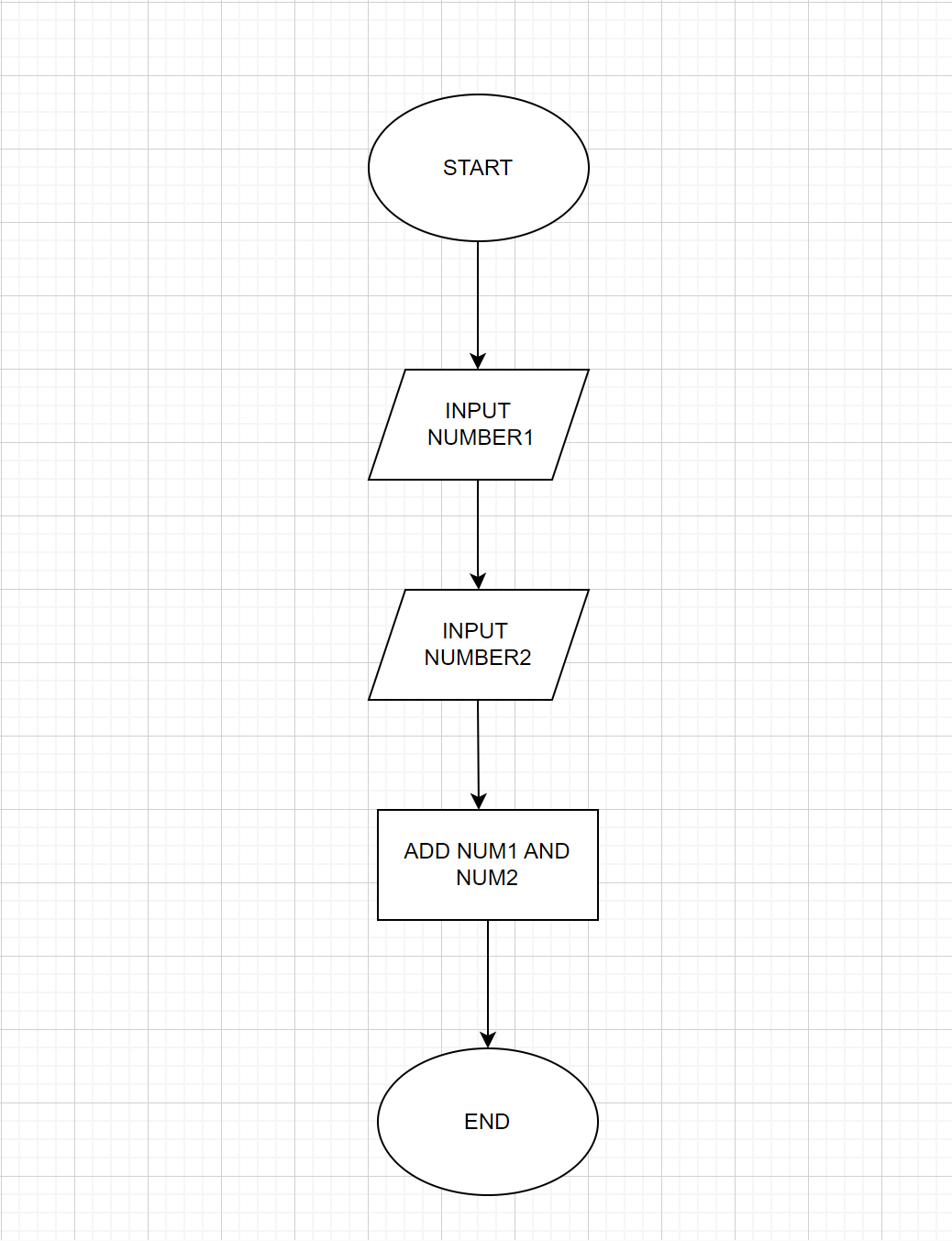




5. What is Flow chart? Create a flowchart to make addition of two numbers.

Ans :-

A flowchart is a graphical representation of a process, system, or algorithm. It uses various symbols and shapes to illustrate the steps involved in completing a task, the sequence of operations, decision points, and the flow of control or data within the process. Flowcharts are widely used in various fields such as computer programming, business process management, engineering, and education to visually depict complex processes in a clear and understandable manner.



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

Ans :-

A use case diagram is a graphical representation of interactions between actors (users or external systems) and a system, showcasing various ways the system can be utilized. It helps in understanding the functional requirements of a system from the user's perspective.

