Que (1) what is SDLC?

- SDLC is Define As Software Development Life Cycle
- SDLC is Step by Step Process in which we've to deliver or to Develop , Test & Deliver the Software to Customer
- SDLC is a structure imposed on the development of a software product that defines the process for planning, implementation, testing, documentation, deployment, and ongoing maintenance and support.
- SDLC have following Phases have to follows

Phases	Purpose
Requirement Analysis	Collection of Data regarding Customer Requirement
Design & Development	Designer Uses the Software to design as per Client Requirement
Implementation	After Completion of Documentation , Coding part Started to Actual Software Develop
Testing	Checking Validation of Product or Software as per Client Requirement or not
Maintenance	Software Placed in Actual Work Environment and then after any modification regarding Defects or new implement are placed in this phase

(I) Requirement Gathering & Analysis

- The Document is generated by Product Manager/Team Leader in the form of SRSD (Software Requirement Specification Document)
- In this SRS Document, all the Details Regarding Project (like Specification, Duration, Budget, etc) are mentioned which is collected from Customer/Client side

(II) Design Phase

- Accordingly SRS Document, Design Team Involved
- Design Team Designed HLD (High Level Design) and LLD (Low Level Design) as per entire requirement for development process of Software or Product

(III) Implementation Phase

- Accordingly this phase, the Actual Software or Product Development done
- In this Phase, Developer coding for Product or Software

(IV) Testing Phase

- In this phase actual software tester involved
- Tester will test the software or product as per client requirement after completion of development of Software

(V) Maintenance Phase

- In this phase, after tester testing software, the software will deploy to customer environment (means software install in customer environment
- After starting using the software or product requiring updating or some improvement in features are considering maintenance phases

Que (2) what is software testing?

- Testing is Define for releasing quality product to customer
- Testing is Part of Development Process
- Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not
- Software testing is a process used to identify the correctness, completeness, and quality of developed computer software
- It can also be stated as the process of validating and verifying that a software program/application/product
- Testing Which is follows requirement of client (like Business and Technical requirement) that guided it's design and development team
- Works as per expected by Developer/Tester/Client
- Definition of software testing have following parts

(i) Process

- Testing is a process rather than single activity

(ii) All Life Cycle Activities

- Testing is part of SDLC process
- The Process of designing test initially to prevent defects which is introduced in the code
- Which is referred as "verifying the test basis (Document includes requirement and design specification) via the test design"

(iii) Static Testing

- Static Testing is process for reviewing of the document which is prepared by Product Manager/Team Leader
- Reviewing means to ensure the document is correct or complete correctness and completeness of the document will be tested in the form of review
- Reviewing term also consist of other three things, which we called as
- Reviewing
- Walkthrough
- Inspection

(iv) Dynamic Testing

- Dynamic testing is process for testing of the Software code which is performed by Software Developer
- Dynamic testing is testing of actual software
- Dynamic testing classified in four level of testing

(1) Unit Testing (3) System Testing
(2) Integration Testing (4) UAT testing (User Acceptance Testing)

(v) Planning

- Planning requires for testing product/software

- Planning is required for controlling test activities, test progress report and status of software under test

(vi) Preparation

- It is requires for doing right selection for testing condition and designing test cases

(vii) Evaluation

- Evaluating the Test is for reviewing the result and progress of test which is under estimated about testing criteria for performing test

(viii) Software Product and related work products

- Along with the testing of code the testing of requirement and design specification and also the related document like operation, user and training material is equally important

Que (3) what is agile methodology?

- Agile SDLC model is a combination of iterative and incremental process model with focus on process adaptability and customer satisfaction by rapid delivery of working software product
- Agile Method break the product into small incremental builds
- These builds are provided in iterations
- Each iteration typically lasts from about one to three weeks
- Every iteration involves cross functional teams working simultaneously on various area like planning, requirements analysis, designing, coding, unit testing, acceptance testing
- At the end of the iteration a working product is displayed to the customer and important stakeholders
- Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes (small time frames) to deliver specific features for a release
- Iteration approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features, the final build holds all the features required by the customer
- Agile though process had started early in the software development and started becoming popular with time due to its flexibility and adaptability

1. **PROs for Agile Model**

- Realistic approach to software development
- Promotes teamwork and cross training
- Functionality can be developed rapidly and demonstrated
- Minimum number of resources required
- Suitable for fixed/changing requirement
- Deliver early partial working solutions
- Good model for environment that change steadily
- Minimum rules, documentation easily employed

- Easy to manage
- No or minimum planning required
- Gives flexibility to developers

2. <u>CONs for Agile Model</u>

- Not useful for complex dependencies
- More risk of sustainability, maintainability and extensibility
- Daily Practise required for operating Agile model by Agile Leader or PM
- Strict Delivery management dictates the scope, functionality to be delivered and adjustments to meet deadlines
- Depends heavily on customer interaction because if not clarification by customer then team will leads wrong directions
- Highly individual dependency because of minimum documentation generated
- Transfer of technology to new team member may be quite challenging due to lack of documentation

Que (4) what is SRS?

- SRS is Define As Software Requirement Specification
- SRS Document generated based on Customers/Client Requirement and Technical Aspects
- It stands for complete description of the behaviour of the system to be developed
- It includes a set of use cases that describes all of the interactions that the users will have with the software
- Use cases are also known as functional requirement, in addition to use cases, the SRS also contains non-functional (or supplementary) requirements
- Non Functional requirements are requirements which impose constrains on the design or implementation (such as performance requirements, quality standards, or design constrains)
- Recommended approaches for the specification of software requirements are described by IEEE 830-1998
- This standard describes possible structures, desirable contents, and qualities of a software requirements specifications
- Types of Requirement
- Customer Requirement
- Functional Requirement
- Non-Functional Requirement
- **Customer Requirement**

➢ The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer.

- **Functional Requirement**

1. Functional Requirements are very important system requirements in the system design process

2. These requirements are the technical specifications, system design parameters and guidelines, data manipulation, data processing, and calculation modules etc, of the proposed system

- **Non-Functional Requirement**

➢ Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system, rather than specific behaviours.

➢ Non-functional requirements are qualities or standards that the system under development must have or comply with, but which are not tasks that will be automated by the system.

Que (5) what is OOPS?

- OOPs is Define As Object Oriented Programming
- Programming is used for Developing Software
- In this type of programming specific programming are used for developing specific purpose
- It's used some particular languages are known as Object Oriented Programming Languages
- An object-based programming language is one which easily supports object-orientation

Smalltalk	Founder of Alan Kay(1972-1980)
C++	Bjarne Stroustrup(1986)
Java(Oak)	Founder of James Gosling
C#	Developed at Microsoft by Anders Hejlsberg et al, 2000
Others	Effile, Objective-C, Ada, ...

Que (6) Write Basic Concepts of oops?

- Identifying objects and assigning responsibilities to these objects.
- Objects communicate to other objects by sending messages.
- Object is called as Black box
- Black box is stands for Hiding of internal details
- Object is derived from abstract data type
- Object-oriented programming has a web of interacting objects, each house-keeping its own state
- Objects of a program interact by sending messages to each other

Que (7) what is object?

- An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain
- An "object" is anything to which a concept applies
- This is the basic unit of object oriented programming(OOP)
- That is both data and function that operate on data are bundled as a unit called as object

- The two parts of an object Object = Data + Methods or to say the same differently
- An object has the responsibility to know and the responsibility to do

Que (8) what is class?

- Class is define for a blueprint or template for creating objects
- It defines the structure and behaviour that object of the class will possess
- Class is breakdown in following terms as Blueprint, Attributes (data), Methods (Behavior), Object Creation, Encapsulation
- Class is define only which action take place on object but it isn't define any data regarding Object that means having attributes that represents the characteristics or properties of the objects
- A class represents an abstraction of the object and abstracts the properties and behaviour of that object
- Class also defines methods, which are functions that define the actions or behaviours that objects of that class can perform
- Objects are instances of a class
- When you create an object, you essentially creating a specific realization of the blueprint defined by the class
- Class can be considered as the blueprint or dentition or a template for an object and describes the properties and behaviour of that object, but without any actual existence
- An object is a particular instance of a class which has actual existence and there can be many objects (or instances) for a class
- Class are key part of encapsulation, which is a major tenant of oop.
- Encapsulation is the bundling of data, and methods that operate on that data, into a single unit

Que (9) what is encapsulation?

- Encapsulation is a fundamental OOP concept that involves bundling data (attributes) and the methods (functions) that operate on that data into a single unit, called a class
- It also involves hiding the internal implementation details of a class and exposing only the necessary interface to the outside world.
- Encapsulation is the practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects
- Encapsulation is placing the data and the functions that works on that data in the same place
- While working with procedural languages, it's not always clear which functions work on which variables but object-oriented programming provides you framework to place the data and the relevant functions together in the same object
- Encapsulation in java is the process of wrapping up of data (properties) and behaviour (method)of an object into a single unit; and the unit here is a class (or Interface)
- Encapsulation in plain English means to enclose or be enclosed in or as if in a capsule
- In java, class is the capsule (or unit)

Que (10) what is inheritance?

- Inheritance means that one class inherits the characteristics of another class. This is also called a “is a” relationship
- In object oriented programming (OOP), inheritance is a powerful mechanism that allows you to create new classes based on existing ones
- Here’ s a breakdown of what that means:
- Core Concept:

❖ Reusability

- The primary goal of inheritance is to promote code reusability
- Instead of writing the same code repeatedly, you can create a base class (also called a superclass or parent class) that contains common attributes and methods
- Then, you can create derived classes (also called subclasses or child classes) that inherit those attributes and methods

❖ Hierarchy

- Inheritance establishes a hierarchical relationship between classes
- This allow you to organize your code in a logical and structured way, reflecting real-world relationships

❖ Extensibility

- Subclasses can extended the functionality of the base class by adding new attributes and methods or by overriding existing ones
- Key Terms

❖ Superclass (parent class)

- The class whose properties and methods are inherited

❖ Subclass (child class)

- The class that inherits properties and methods from the superclass
- How it works:
- A subclass “inherits” the attributes and methods of its superclass
- This means that it automatically has access to those properties and behaviours
- Subclasses can also add their own unique attributes and methods, allowing them to specialize the behaviour of the superclass
- Subclasses can “override” methods from the superclass, providing their own implementation of those methods
- Benefits of inheritance

❖ Code Reduction

- Reduces code duplication, making your code more concise and easier to maintain

❖ Improved Organization

- Helps organize code into logical hierarchies, making it easier to understand and manage

❖ Increased flexibility

- Allows you to easily extend and modify existing code without breaking existing functionality

Que (11) what is polymorphism?

- Polymorphism is a fundamental concept in object-oriented programming (oop)
- Essentially, it means "many forms"
- In the context of programming, it refers to the ability of something to take on multiple forms
- It allows different objects to respond to the same message in different ways, the response specific to the type of the object
- The most important aspect of an object is its behaviour (the things it can do). A behaviour is initiated by sending a message to the object (usually by calling a method)
- Key Concepts:
- Meaning:
  - The Word "Polymorphism" itself comes from the Greek words "poly" (many) and "morph" (form)
  - In Programming, it's the ability of a single action or interface to behave differently depending on the underlying object
- Purpose:
  - Polymorphism promotes code reusability and flexibility
  - It allows you to write more generic code that can work with a variety of objects
- Type of Polymorphism:
  - Comple-time polymorphism (static polymorphism):
    - This occurs during the compilation of the program
    - It's often achieved through:
- Function overloading: Having multiple functions with the same name but different parameters.
- Operator overloading: Giving operators (like +,-,etc.) different meaning depending on the operators
- Runtime Polymorphism (dynamic polymorphism):
  - This occurs during the execution of the program
  - It is primarily achieved through:
    - Method overriding: A subclass providing its own implementation of a method that is already defined in its superclass

Que (12) Draw Usecase on online book shopping.

- Follow this Link to see Usecase diagram for online book shopping

Que (13) Draw Usecase on online bill payment system (paytm).

- Follow this Link to see Usecase diagram for online bill payment system (paytm)

Que (14) Draw Usecase on Online shopping product using COD.

- Follow this Link to see Usecase on Online shopping product using COD

Que (15) Write SDLC phases with basic introduction.

- The Software Development Life Cycle (SDLC) is a process followed for a software project, within a software organization
- It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software
- The life cycle defines a methodology for improving the quality of software and the overall development process
- Here's a breakdown of the typical SDLC phases:

☑ Planning
☑ Requirement Analysis
☑ Design
☑ Coding (Implementation)
☑ Testing
☑ Deployment
☑ Maintenance

☑ Planning

- This Initial phase involves defining the project's scope, goals, and feasibility
- It includes resource allocation, scheduling, and risk assessment

☑ Requirement Analysis

- In this phase, the team gathers and documents the specific requirements of the software
- This involves understanding the needs of the stakeholders and users

☑ Design

- This phase focused on creating the software's architecture and design
- This includes designing the user interface, database, and overall system structure

☑ Testing

- In this phase, the software is thoroughly tested to identify and fix any bugs or defects
- This ensures that the software meets the requirement and functions as expected

☑ Deployment

- This phase involves releasing the software to the users or the production environment

☑ Maintenance

- After deployment, the software needs ongoing maintenance, including bug fixes, updates, and enhancements

Que (16) Explain Phases of the waterfall model.

- The Waterfall model is a classic, linear project management approach where each phase of the project must be completed before the next phase begins
- It's called "Waterfall" because the process flows sequentially downward, like a waterfall
- Here's a breakdown of the typical phases:

☑ Requirements Gathering/Analysis:

- This is the initial phase where the project's goals, objectives, and requirements are defined
- It involves gathering input from stakeholders to understand their needs and expectations
- The outcome of this phase is a detailed requirements document that outlines what the system or product should do

☑ Design:

- In this phase, the system's architecture, design, and specifications are created based on the requirements gathered in the previous phase
- This includes both high-level and low-level design, covering aspects like user interfaces, database structures, and system components

☑ Implementation (Coding):

- This is where the actual development or construction of the system takes place
- Developers write code, build components, and implement the design specifications

☑ Testing:

- Once the implementation phase is complete, the system is thoroughly tested to identify and fix any defects or bugs
- Various testing methods are employed to ensure that the system meets the requirements and functions as intended

☑ Deployment:

- In this phase, the completed system is deployed or installed in the production environment
- This involves making the system available to end-users

☑ Maintenance:

- After deployment, the system enters the maintenance phase, where ongoing support, updates, and bug fixes are provided
- This phase ensures that the system continues to operate effectively and meet the evolving needs of users

☑ Key characteristics of the Waterfall model:

- Linear and Sequential: Each phase must be completed before moving to the next.
- Rigid Structure: Changes to requirements or design can be difficult and costly once a phase is completed.
- Documentation-Heavy: Each phase typically involves extensive documentation

Que (17) Write phases of spiral model.

- The Spiral Model is a software development process that combines elements of both design and prototyping-in-stages, with an emphasis on risk analysis
- It's characterized by its iterative nature, where the project repeatedly cycles through a series of phases
- Here's a breakdown of those phases:

☑ Planning:

- This phase involves defining the project's objectives, alternatives, and constraints
- It includes gathering requirements, understanding project goals, and planning for the next iteration

☑ Risk Analysis:

- This is a crucial phase where potential risks are identified, assessed, and analyzed
- Strategies are developed to mitigate these risks, which might involve prototyping, simulations, or other techniques

☑ Development:

- In this phase, the actual development of the product takes place
- This includes designing, coding, and testing the software, based on the requirements and risk assessments

☑ Review (Evaluation):

- This phase involves evaluating the product and the development process
- Customers provide feedback, and the results are reviewed to determine if the project should proceed to the next iteration
- This phase also contains planning for the next phase

☑ Key aspects of the Spiral Model:

- Iterative: The process repeats through the four phases, with each iteration refining the product
- Risk-driven: Risk analysis is a central focus, influencing decisions at each stage
- Flexible: The model can adapt to changing requirements and risks

Que (18) Write agile manifesto principles.

- The Agile Manifesto is a foundational document for software development, and it's built upon a set of core principles
- Here's a breakdown:

☑ The Agile Manifesto Values:

- It's important to first understand the 4 values that the manifesto is based upon

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

☑ The 12 Agile Principles:

- These principles expand upon the values and provide guidance for implementing agile practices. Here are the 12 principles:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale

4. Business people and developers must work together daily throughout the project
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation
7. Working software is the primary measure of progress
8. Agile processes promote sustainable development
- The sponsors, developers, and users should be able to maintain a constant pace indefinitely
9. Continuous attention to technical excellence and good design enhances agility
10. Simplicity–the art of maximizing the amount of work not done–is essential
11. The best architectures, requirements, and designs emerge from self-organizing teams
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly
- These principles emphasize collaboration, flexibility, and continuous improvement, which are key to agile software development

Que (19) Explain working methodology of agile model and also write pros and cons.

- The Agile methodology is a dynamic and iterative approach to project management, primarily used in software development
- Here's a breakdown of its working methodology, along with its pros and cons:

☑ Working Methodology:

- Iterative Development:
  - Agile breaks down projects into small, manageable iterations called "sprints."
  - Each sprint typically lasts a few weeks
  - At the end of each sprint, a working version of the software is delivered
- Customer Collaboration:
  - Agile emphasizes continuous interaction with the customer or stakeholders
  - Feedback is gathered throughout the development process, allowing for adjustments and improvements
- Adaptive Planning:
  - Instead of rigid, upfront planning, Agile embraces flexibility
  - Plans are adapted based on feedback and changing requirements.
- Self-Organizing Teams:
  - Agile teams are typically cross-functional and self-organizing
  - Team members collaborate closely and take ownership of their work.
- Continuous Improvement:
  - Agile promotes regular reflection and improvement
  - Teams hold retrospective meetings to identify areas for improvement

☑ **Pros:**

- Increased Flexibility:
  - Agile's adaptability allows for quick responses to changing requirements

- Enhanced Customer Satisfaction:
  - Continuous customer involvement ensures that the final product meets their needs
- Improved Product Quality:
  - Frequent testing and feedback lead to higher-quality software
- Faster Time to Market:
  - Iterative development allows for quicker delivery of working software
- Improved Team Collaboration:
  - Agile fosters strong communication and collaboration among team members
- Reduced Risk:
  - By having frequent releases, and feedback, the risk of building the wrong product is greatly reduced

☑ **Cons:**

- Lack of Predictability:
  - Agile's flexibility can make it difficult to predict project timelines and budgets
- Scope Creep:
  - Continuous changes can lead to expanding project scope
- Documentation Challenges:
  - Agile prioritizes working software over extensive documentation
- Requires Strong Customer Involvement:
  - Agile relies on active customer participation, which may not always be possible
- Scaling Challenges:
  - Implementing Agile in large, complex projects can be difficult
- Dependency on Team Dynamics:
  - A teams success is very dependent on the teams ability to communicate, and work together

In essence, Agile is best suited for projects that require flexibility, collaboration, and continuous improvement

Que (20) Draw Usecase on Online shopping product using payment gateway.

- Follow this Link to see Usecase diagram for Online shopping product using payment gateway