1. **What is SDLC ?**

* 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.

1. **What is software testing ?**

* Software testing is a process used to identify the correctness completeness and quality of developed computer software.

1. **What is agile methodology ?**

* Agile SDLC model is a combination of interative and incremental process models with Focus on process adaptability and customer satisfaction by rapid delivery of working Softwaer product.

1. **What is SRS ?**

* Software Requirements Specification. It's a detailed document that defines the

requirements for a software system, including both functional and non-functional

aspects. This document is a crucial reference for developers, testers, and other

stakeholders during the software development life cycle.

1. **What is oops ?**

* Object – oriented programing is a programing paradigm that uses “object” to design Softwer. It is based on several key concepts that help in organizing code and making it more modular, reusable and easier to maintain.

1. **Write Basic Concepts of oops ?**

* Classes and Objects
* Encapsulation
* Inheritance
* Polymorphism = Overloading, Overriding3
* Abstraction

1. **What is object ?**

* A blueprint or template for creating objects. It defines a set of attributes (data) and methods (functions) that the created objects will have.

1. **What is class ?**

* An instance of a class. It represents a specific entity with the properties and behaviors defined by its class.

1. **What is encapsulation ?**

* Encapsulation is the bundling of data (attributes) and methods(functions) that operate on the data into a single unit, or class. It restricts direct access to some of the object’s components, which is a means of preventing unintended interference and misuse of the methods and data.
* Access modifiers ( like private, protected, and public) are used to control the visibility of class members.

1. **What is inheritance ?**

* Inheritance is mechanism that allows one class ( the child or subclass) to inherit the attributes and methods of another class (the parent or superclass). This promotes code reusability and establishes a hierarchical relationship between classes.
* For example, if you have a class Animal, a class Dog can inherit from Animal, gaining its properties and behaviors while also adding its own.

1. **What is polymorphism ?**

* Polymorphism allows methods to do different things based on the object it is acting upon, even though they share the same name. This can be achieved through:
* **Method Overloading** :  Same method name with different parameters within the same class.
* **Method Overriding** : A subclass provides a specific implementation of a method that is already defined in its superclass.

1. **Draw Usecase on online bill payment system (paytm)**

* [**https://drive.google.com/file/d/1UTWad8g4BQTHCH9HHDTuhy-d2KWG8Mjw/view?usp=sharing**](https://drive.google.com/file/d/1UTWad8g4BQTHCH9HHDTuhy-d2KWG8Mjw/view?usp=sharing)

1. **Draw Use case on banking system for customers.**

* [**https://drive.google.com/file/d/1v6xEV6cLReLKtAL0UFBVBAlDx9rWKfJF/view?usp=sharing**](https://drive.google.com/file/d/1v6xEV6cLReLKtAL0UFBVBAlDx9rWKfJF/view?usp=sharing)

1. **Draw Usecase on Broadcasting System.**

* [**https://drive.google.com/file/d/14tW7OnsdJAxo5Vlgd9Ylas1ROTuS4IOM/view?usp=sharing**](https://drive.google.com/file/d/14tW7OnsdJAxo5Vlgd9Ylas1ROTuS4IOM/view?usp=sharing)

1. **Write SDLC phases with basic introduction.**

**SDLC**  is structure imposed in the development of a software product that defines the process for planning, implementation, testing, documentation, deployment, and

Ongoing maintenance and support.

* A software development life cycle is essentially a series of steps, or phases, that provide a model of the development and lifecycle management of an application or piece of software.
* The methodology within the SDLC process can vary across industries and organizations, but standards such as ISO/ IEC12207 represent processes that establish a lifecycle for software, and provide a mode for the development, acquisition, and configuration of software systems.

**SDLC Phases**

|  |  |
| --- | --- |
| Requirements  Collection/Gathering | Establish Customer Needs |
| Analysis | Model and specify the requirements- “what” |
| Design | Model and specify a solution – “Why” |
| Implementation | Construct a Solution in software |
| Testing | Validate the solution against the requirements |
| Maintenance | Repair defects and adapt the solution to the new requirements |

**Requirements :**  In this phase, detailed requirements are collected from stakeholders. This includes functional and non-functional requirements. Analysts work to understand what the users need from the software, which will guide the design and development process.

**Analysis :** The analysis phase of the Software Development Life Cycle (SDLC) is a critical step that focuses on understanding and documenting the requirements of the software project. This phase serves as a bridge between the initial planning and the subsequent design and implementation phases.

**Design**:  The design phase translates the requirements into a blueprint for the software. This includes architectural design, user interface design, and database design. The goal is to create a detailed specification that developers can follow during implementation.

**Implementation :** During this phase, developers write the actual code based on the design specifications. This is where the software is built, and it often involves multiple iterations and collaboration among team members.

**Testing**: After implementation, the software undergoes rigorous testing to identify and fix defects. Various testing methods, such as unit testing, integration testing, and user acceptance testing, are employed to ensure the software functions as intended and meets the requirements.

**Maintenance:** After deployment, the software enters the maintenance phase, where it is monitored for issues, and updates or enhancements are made as needed. This phase ensures the software remains functional and relevant over time.

1. **Explain Phases of the waterfall model**

* The waterfall model is a linear, sequential approach to project management and software development, where each phase is completed before moving to the next. It consists of distinct stages requirements analysis, system design, implementation, testing, deployment, and maintenance – with clear deliverables at each step. Progress flows in one direction, like a waterfall, without overlapping or iterating between phases. It is best suited for projects with well – defined stable requirements but can be inflexible if changes are needed.

1. **Requirements Analysis**

* All project requirements are gathered and documented.
* Stakeholders and clients define what the system should do (functional and non-functional requirements).
* Output : A detailed requirements specification document.

1. **System desing**

* The system’sarchitecture and technical specifications are created based on the requirements.
* Includes designing hardware, software, databases, and user interfaces.
* **Output:**  design documents ( e.g., architecture diagrams, data models).

1. **Implementation (Coding)**

* Developers write code to build the system based on the design specifications.
* The system is broken into smaller modules or unit for coding.
* Output : Functional software or system components.

1. **Testing.**

* The system is tested to ensure it meets the requirements and is free of defects.
* Includes unit testing, integration testing, system testing, and acceptance testing.
* Output : Bug reports and a verified system.

1. **Deployment**

* The tested system is deployed to the production environment for end – use.
* May involve installation, configuration, and user training.
* Output : A fully operational system.

1. **Maintenance**

* Ongoing support to fox bugs, update the system, or add new features.
* Ensures the system remains functional and relevant.
* Output : Updated system versions or patches.

1. **Write phases of spiral model.**

* Spiral model is very widely used in the software industry with the natural development process of any product. I.E learning with maturity and also involves minimum risk for the customer as well as the development firms.

1. **Planning :**

* In this phase, the objectives of the project are defined. This includes identifying the requirements, constraints, and risk associated with the project. Stakeholders are involved to gather their input and expectations.
* A feasibility study is conducted to assess whether the project is viable in terms of technical, operational, and economic aspects.
* The project scope is outlined, and a preliminary schedule and budget are established.

1. **Risk Analysis Phase:**

* This phase focuses on identifying and analyzing potential risks that could impact the project. Risks can be technical, financial, or related to project management.
* Various risk mitigation strategies are developed to address the identified risks. This may involve creating prototypes or conducting simulations to better understand the risks.

1. **Engineering Phase :**

* In this phase, the actual development of the software takes place. This includes designing, coding, and testing the software components**.**
* Prototypes may be developed to validate requirements and gather feedback from stakeholders. This iterative process allows for adjustments based on user feedback.
* The engineering phase may involve multiple iterations, with each iteration producing a refined version of the software..

1. **Evaluation Phase:**

* After the engineering phase, the developed software is evaluated against the initial requirements and objectives. This includes testing the software for functionality, performance, and reliability**.**
* Stakeholders review the software to ensure it meets their expectations and requirements. Feedback is collected for further improvement

1. **Write agile manifesto principles.**

* The Agile Manifesto, created in 2001 by a group of software developers, outlines four fundamental values and twelve principles that guide Agile software development. Here are the twelve principles along with a brief explanation of each:

1. **Customer Satisfaction:**

* Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
* This principle emphasizes the importance of delivering functional software frequently to meet customer needs and expectations.

1. **Welcome Changing Requirements**

* Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
* Agile methodologies embrace change, allowing teams to adapt to new requirements and market conditions, ensuring that the final product is relevant and valuable.

1. **Deliver Working Software Frequently:**

* Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
* Regular delivery of functional software helps to maintain momentum and allows for early feedback from stakeholders.

1. **Collaboration Between Business and Developers**

* Business people and developers must work together daily throughout the project.
* Close collaboration fosters better understanding and communication, leading to more effective solutions that meet business needs.

1. **Motivated Individuals**

* Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
* Empowering team members and providing them with the necessary resources and autonomy leads to higher productivity and job satisfaction.

1. **Face-to-Face Conversation:**

* The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
* Direct communication reduces misunderstandings and fosters a collaborative environment.

1. **Working Software is the Primary Measure of Progress:**

* Working software is the primary measure of progress.
* This principle shifts the focus from traditional metrics (like documentation) to the actual functionality delivered, ensuring that the team is aligned with customer needs.

1. **Sustainable Development:**

* Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
* This principle emphasizes the importance of a balanced workload to prevent burnout and ensure long-term productivity.

1. **Technical Excellence and Good Design:**

* Continuous attention to technical excellence and good design enhances agility.
* High-quality code and design practices lead to more maintainable and adaptable software, facilitating future changes.

1. **Simplicity:**

* The art of maximizing the amount of work not done is essential.
* This principle encourages teams to focus on the most valuable features and avoid unnecessary complexity, leading to more efficient development.

1. **Self-Organizing Teams:**

* The best architectures, requirements, and designs emerge from self-organizing teams.
* Empowering teams to organize themselves fosters creativity and innovation, leading to better solutions.

1. **Regular Reflection and Adjustment:**

* At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

1. **Explain working methodology of agile model and also write pros and cons.**

The Agile model is a flexible and iterative approach to software development that emphasizes collaboration, customer feedback, and rapid delivery of functional software. Here’s an overview of its working methodology, along with its pros and cons.

**Working Methodology of Agile Model**

1. **Iterative Development:**

* Agile development is divided into small, manageable units called iterations or sprints, typically lasting from one to four weeks. Each iteration involves planning, design, coding, testing, and review.

1. **User Stories:**

* Requirements are captured as user stories, which describe features from the end-user's perspective. This helps prioritize work based on customer value.

1. **Daily Stand-ups:**

* Teams hold daily stand-up meetings (or scrums) to discuss progress, challenges, and plans for the day. This fosters communication and quick problem-solving.

1. **Collaboration:**

* Agile emphasizes collaboration among cross-functional teams, including developers, testers, and business stakeholders. This ensures that everyone is aligned and can contribute to the project.

1. **Continuous Feedback:**

* After each iteration, teams demonstrate the working software to stakeholders to gather feedback. This allows for adjustments based on user input and changing requirements.

1. **Retrospectives:**

* At the end of each iteration, teams conduct retrospectives to reflect on what went well, what didn’t, and how processes can be improved. This promotes a culture of continuous improvement.

1. **Incremental Delivery:**

* The goal is to deliver functional software at the end of each iteration, allowing users to start using parts of the system early in the development process.
* Agile allows teams to respond quickly to changes in requirements, making it easier to adapt to evolving customer needs and market conditions.

**Pros of Agile Model.**

1. **Flexibility and Adaptability:**

* Agile allows teams to respond quickly to changes in requirements, making it easier to adapt to evolving customer needs and market conditions.

1. **Customer Engagement:**

* Regular feedback from customers ensures that the final product aligns with their expectations, leading to higher satisfaction.

1. **Faster Time to Market:**

* By delivering working software in short iterations, Agile enables quicker releases, allowing businesses to capitalize on opportunities sooner.

1. **Improved Quality:**

* Continuous testing and integration throughout the development process help identify and fix issues early, resulting in higher-quality software.

1. **Enhanced Team Collaboration:**

* Agile promotes a collaborative environment, fostering better communication and teamwork among members, which can lead to innovative solutions.

1. **Focus on Value:**

* Prioritizing user stories based on customer value ensures that the most important features are developed first, maximizing the return on investment.

**Cons of Agile Model**

* **.Less Predictability**:
* The flexible nature of Agile can lead to uncertainty in project timelines and budgets, making it challenging to predict the final outcome.
* **Requires Cultural Shift:**
* Implementing Agile may require significant changes in organizational culture and mindset, which can be met with resistance from team members accustomed to traditional methodologies.
* **. Scope Creep:**
* The openness to changing requirements can lead to scope creep if not managed properly, potentially overwhelming the team with too many changes.
* **Documentation Challenges**:
* Agile emphasizes working software over comprehensive documentation, which can lead to insufficient documentation for future reference or onboarding new team members.

1. **Draw use case on OTT Platform.**

* [**https://drive.google.com/file/d/1tc5rMAJRLt17VMcmnT9eqmO4Gm7UG1vg/view?usp=sharing**](https://drive.google.com/file/d/1tc5rMAJRLt17VMcmnT9eqmO4Gm7UG1vg/view?usp=sharing)

1. **Draw usecase on E-commerce application.**

* [**https://drive.google.com/file/d/1CxiqDL\_jFaJXCulRZwgMRSgpMG6S-G2a/view?usp=sharing**](https://drive.google.com/file/d/1CxiqDL_jFaJXCulRZwgMRSgpMG6S-G2a/view?usp=sharing)

1. **Draw usecase on Online shopping product using payment gateway.**

* [**https://drive.google.com/file/d/1-vmPLq7-G\_t-S\_V5a0QT4tpoFWIjRneY/view?usp=sharing**](https://drive.google.com/file/d/1-vmPLq7-G_t-S_V5a0QT4tpoFWIjRneY/view?usp=sharing)