1.What is SDLC?

SDLC stands for Software Development Life Cycle.

The SDLC is a systematic sequential approach to software development. The cost-effective and time-efficient process that development teams use to design and build high-quality software.

- Planning /Requirement
- Analysis
- Designing
- Implementation
- Testing
- Maintenance

2.What is software testing?

An approach to develop any product with high quality and lowest cost within the shortest possible time.

- There are two types of software testing

Manual Testing

Manual testing is a process of executing test cases manually by QA without any automation tool.

- accuracy about finding defects 100% can be possible in manual testing.

Automation Testing

Automation testing is a process of executing test cases by QA with automation tool.

- Accuracy about finding defects 100% not possible in automation testing.

3.What is agile methodology?

Agile methodology is a project management and software development approach that emphasises flexibility, collaboration, and customer-centricity. It promotes iterative progress through small, incremental changes rather than following a strict, linear process.

Key principles of Agile include:
-Customer Collaboration: Regularly engaging with customers to gather feedback and adjust the product based on their needs.
-Iterative Development: Breaking projects into small, manageable units (sprints) that allow teams to deliver functional features quickly.
-Cross-Functional Teams: Encouraging collaboration among team members with diverse skills, promoting effective communication and problem-solving.
-Adaptive Planning: Being open to changes in requirements even late in the development process to better meet customer needs.
-Continuous Improvement: Regularly reflecting on processes and outcomes to identify areas for improvement.

4.What is SRS

SRS stands for Software Requirements Specification. It is a comprehensive document that outlines the functional and non-functional requirements for a software application. The SRS serves as a guide for both development and testing teams, ensuring everyone understands what the software should do and the constraints it operates under.

5.What is oops

Object Oriented Programming is viewed as a collection of objects. It is used to structure the software program into simple reusable code.

6.Write Basic Concepts of oops

- Class: A blueprint for creating objects. It defines properties and behaviors. **Data members member function**

- Object: An instance of a class. It encapsulates data and functions that operate on that data.

- Inheritance: A mechanism that allows one class to inherit the attributes and methods of another class. Promotes code reuse and establishes a relationship between classes.

- Encapsulation: Bundles the data (attributes) and methods (functions) that operate on the data into a single unit (class).Protects the internal state of an object from outside interference and misuse, often using access modifiers (public, private, protected).
- Inheritance:A mechanism that allows one class (subclass or derived class) to inherit the attributes and methods of another class (superclass or base class).Promotes code reuse and establishes a relationship between classes.**security protect the data**

- Polymorphism: The ability of different classes to be treated as instances of the same class through a common interface. It allows methods to do different things based on the object that it is acting upon. This is often implemented through method overriding (in subclasses) and method overloading (same method name with different parameters). **One name having many forms**

- Abstraction:The concept of hiding complex implementation details and showing only the essential features of the object.This is often achieved through abstract classes and interfaces.

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. Java

8.What is class

When you define a class, you define a blueprint for an object. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

9.What is encapsulation

Bundles the data (attributes) and methods (functions) that operate on the data into a single unit (class).Protects the internal state of an object from outside interference and misuse, often using access modifiers (public, private, protected).

- Inheritance:A mechanism that allows one class (subclass or derived class) to inherit the attributes and methods of another class (superclass or base class).Promotes code reuse and establishes a relationship between classes

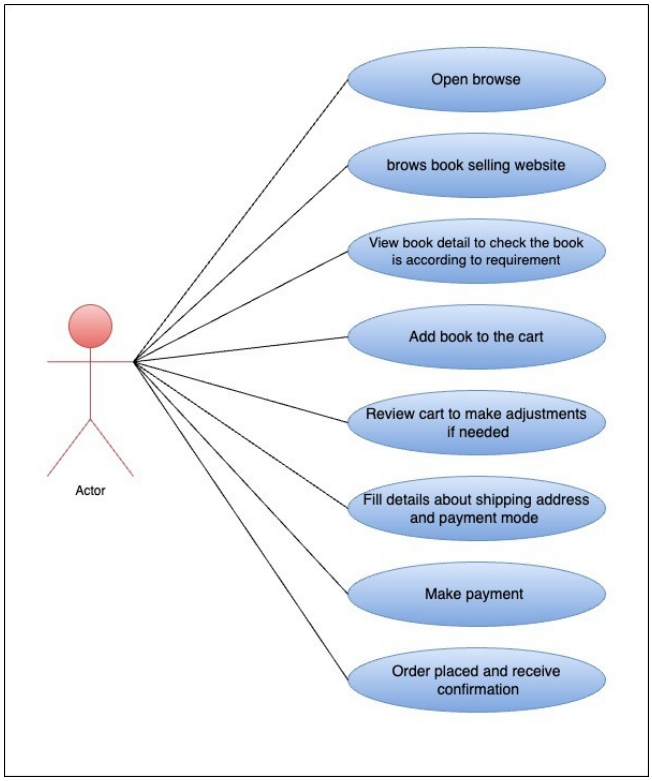
10.What is inheritance

A mechanism that allows one class to inherit the attributes and methods of another class. Promotes code reuse and establishes a relationship between classes.

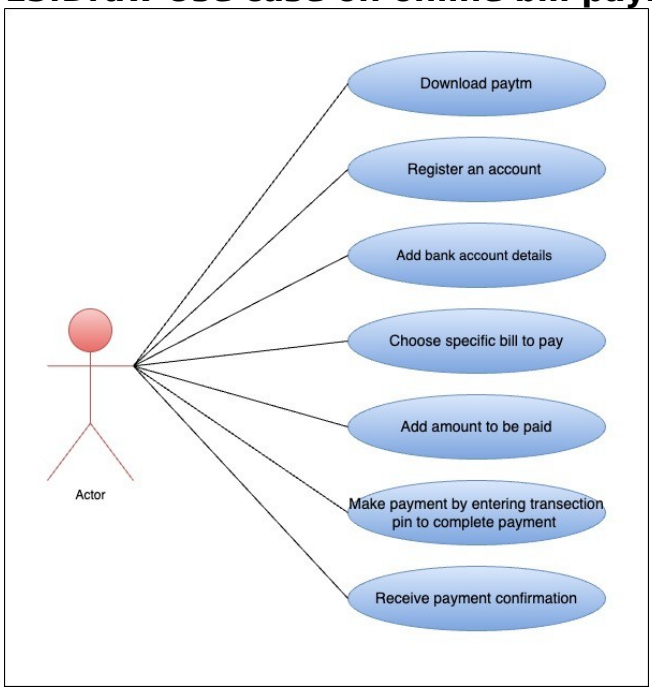
11.What is polymorphism

The ability of different classes to be treated as instances of the same class through a common interface. It allows methods to do different things based on the object that it is acting upon. This is often implemented through method overriding (in subclasses) and method overloading (same method name with different parameters).

12.Draw Use case on Online book shopping



13.Draw Use case on online bill payment system (Paytm)



14.Write SDLC phases with basic introduction

SDLC stands for Software Development Life Cycle.

The SDLC is a systematic sequential approach to software development. The cost-effective and time-efficient process that development teams use to design and build high-quality software.

- Planning /Requirement
- Analysis
- Designing
- Implementation
- Testing
- Maintenance

Planning:

By defining clear roles, responsibilities, and expectations, it lays a solid foundation for an efficient software development process.

- Planning includes requirement gathering and after getting all information there may be problems because of not understanding the clear view of project and create some confusion about requirement by not conforming it also there may be a confusion about requirement amalgamation because of several requirements may be expressed together

-Planning involves defining the software's purpose.The team collaborates to understand the end-users' needs and the goals the software should meet.

-Functional requirement….describe system services or functions.
-Non functional requirement …are constraints (limitation on making solution)on the system or the development process.

Analysis

After collecting the data, the team analyse it and analysis helps the team understand the software's functionality, performance, security, and interface needs

- Analysis includes how to go with the planning and complete behaviour of the system from the data

SRS: software requirement specification
-functional requirement.…describe system services or functions.
-non functional requirement …are constraints (limitation on making solution)on the system or the development process.

Design
The Design phase is all about building the framework and also includes includes implementation plan.
- It's an essential step in creating software that works efficiently and provides an excellent user experience.

Coding
The Coding phase in the Software Development Life Cycle (SDLC) is when engineers and developers start converting the software design into code.

- Developers use an appropriate programming language, Java or otherwise, to write the code, guided by coding guidelines.
- This document acting as a roadmap ensures the software aligns with the vision set in earlier phases.
- The implementation phase deals with issues of quality, performance, baselines, libraries, and debugging

Testing
Quality is very important.Quality is a distinguishing attribute of a system indicating the degree of excellence.
- The testing phase is a separate phase which is performed by a different team after the implementation is completed.

Maintenance
- After the product is released in the market, its maintenance is done for the existing customer base.

15.Explain Phases of the waterfall model

The Waterfall Model provides a structured approach to software development with clear phases and documentation. It is one of the earliest models introduced for software development and is often used in projects with well-defined requirements.
Earlier this model was very popular but nowadays it is not used. However, it is very important because all the other software development life cycle models are based on the classical waterfall model

The waterfall model involves a sequential approach to software development, where each phase of the project is completed before moving on to the next one.
The waterfall model relies heavily on documentation to ensure that the project is well-defined and the project team is working towards a clear set of goals.
The waterfall model places a high emphasis on quality control and testing at each phase of the project, to ensure that the final product meets the requirements and expectations

1. Requirements: The first phase involves gathering requirements from clients and analysing them to understand the scope and objectives of the project.

2. Design: Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.

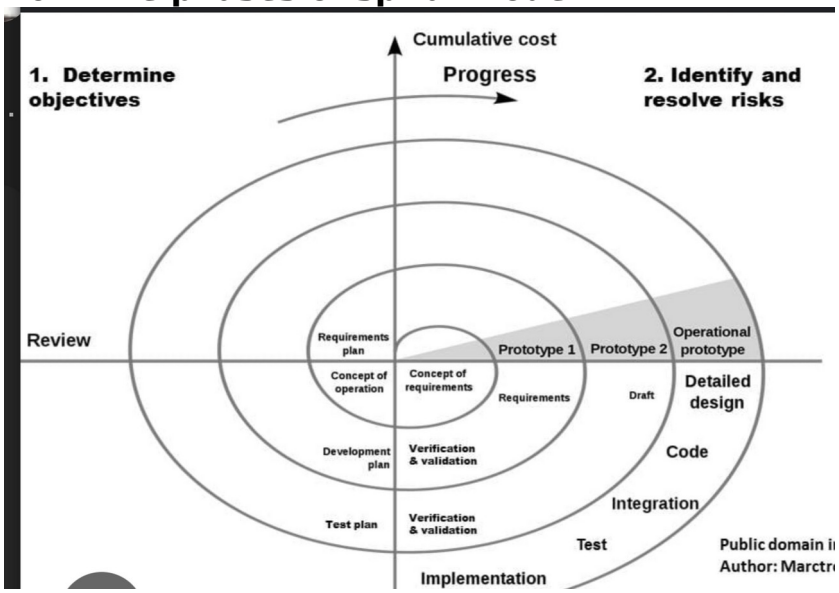
3. Development: The Development phase include implementation involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.

4. Testing: In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.

5. Deployment: Once the software has been tested and approved, it is deployed to the production environment.

6. Maintenance: The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.

16.Write phases of spiral model



Spiral Model is very widely used in the software industry as it is in synch with the natural
development process of any product learning with maturity also involves minimum risk for the
customer as well as the development firms. It provides support for **Risk Handling**.

-Planning: In first phase of the spiral model we clarify what the project aims to achieve, including functional and non-functional requirements.

-Risk Analysis: In the risk analysis phase, the risks associated with the project are identified and evaluated.

-Engineering: In the engineering phase, the software is developed based on the requirements gathered in the previous iteration.

-customer Evaluation: In the evaluation phase, the software is evaluated to determine if it meets the customer's requirements and if it is of high quality.

17.Write agile manifesto principles

-Individuals and interactions over processes and tools
-Working software over comprehensive documentation
-Customer collaboration over contract negotiation
-Responding to change over following a plan

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

Agile methodology is a project management and software development approach that emphasises flexibility, collaboration, and customer-centricity. It promotes iterative progress through small, incremental changes rather than following a strict, linear process.

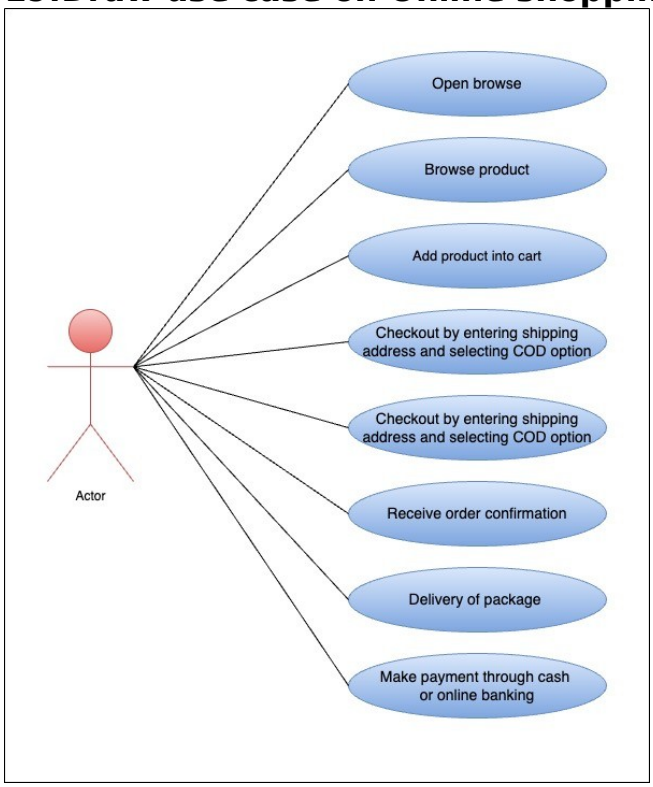
Key principles of Agile include:
-Customer Collaboration: Regularly engaging with customers to gather feedback and adjust the product based on their needs.
-Iterative Development: Breaking projects into small, manageable units (sprints) that allow teams to deliver functional features quickly.
-Cross-Functional Teams: Encouraging collaboration among team members with diverse skills, promoting effective communication and problem-solving.
-Adaptive Planning: Being open to changes in requirements even late in the development process to better meet customer needs.
-Continuous Improvement: Regularly reflecting on processes and outcomes to identify areas for improvement.

Pros:
1. Very realistic approach
2. Rapid delivery.
3. Functionality can be developed rapidly
4. Resource requirements are minimum.
5. Little or no planning required
6. Promotes teamwork and cross training.
7. Suitable for fixed or changing requirements
8. Gives flexibility to developers

Cons:
1. More risk of sustainability, maintainability and extensibility.
2. Depends heavily on customer interactions.
3. Very high individual dependency.
4. Minimum documentation generated.

5. Not useful for small projects.
6. Not suitable for handling complex dependencies.

19.Draw use case on Online shopping product using COD.



20.Draw use case on Online shopping product using payment gateway.