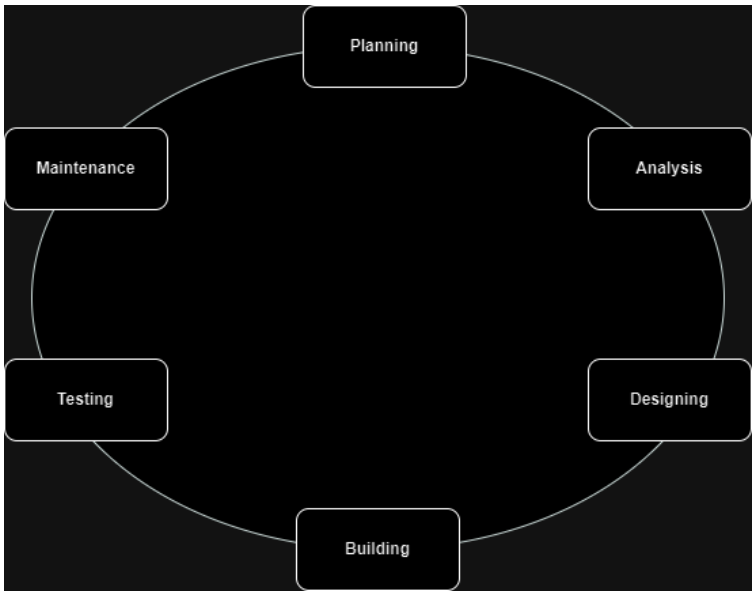
Name: Yagnam Parmar

Course: Diploma in Software Testing & Automation

Assignment: Module 1 (Introduction And fundamental)

Module 1(Introduction and fundamental)

1.What is SDLC?



SDLC is A Step-by-step Approach to develop any Software/With the Lowest Cost and Highest Possible Quality in the Shortest Amount of Time

2. Write SDLC phases with basic introduction?

SDLC has a total of six Phases which are described below.

1.Planning: Problems can be raised while gathering the requirements

--> Lack of clarity

--> Requirement confusion (functional/ non-functional)

--> Requirement Amalgamation (group)

2.Analysis/Defining:

This phase starts with the requirement document delivered by the requirement phase and maps the requirements into architecture

The architecture defines the components, their interfaces and behaviors.

3.Design:

Design Architecture Document

The requirement document must guide this decision process.

The architecture team also converts the typical scenarios into a test plan.

The Design team can now expand upon the information established in the requirement document.

4. Implementation/ Building/ coding:

Software can be Implementation by the technology Language like (java.paython.Ruby.PHP)

The implementation phase deals with issues of quality, performance, baselines, libraries, and debugging.

In the implementation phase, the team builds the components either from scratch or by composition.

5.Testing:

Software testing is a process which is used to identify the correctness, completeness, quality of the developed software.

It is much easier to explain to a customer why there is a missing feature than to explain to a customer why the product lacks quality

6.Maintenance:

Three Types of maintenance

-->Corrective maintenance: identifying and repairing defects

-->Adaptive maintenance: adapting the existing solution to the new platforms.

--> Perfective Maintenance: implementing the new requiremen

3.What is Software Testing?

Software testing is a process which is used to identify the correctness, completeness, quality of the developed software.

In simple words testing is executing a system to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements

. Types of Software:

1) System Software (in-built s/w)

Provided by the system only, No require to install/download

e. g calendar, clock, calculator, notepad,

2) Application Software

Provided by the developer. Need to install or download from play store.

e. g, WhatsApp, Instagram, LinkedIn, Snapchat, Spotify,

form of app, s/w: Web application, desktop application, mobile application

---

4. What is SRS

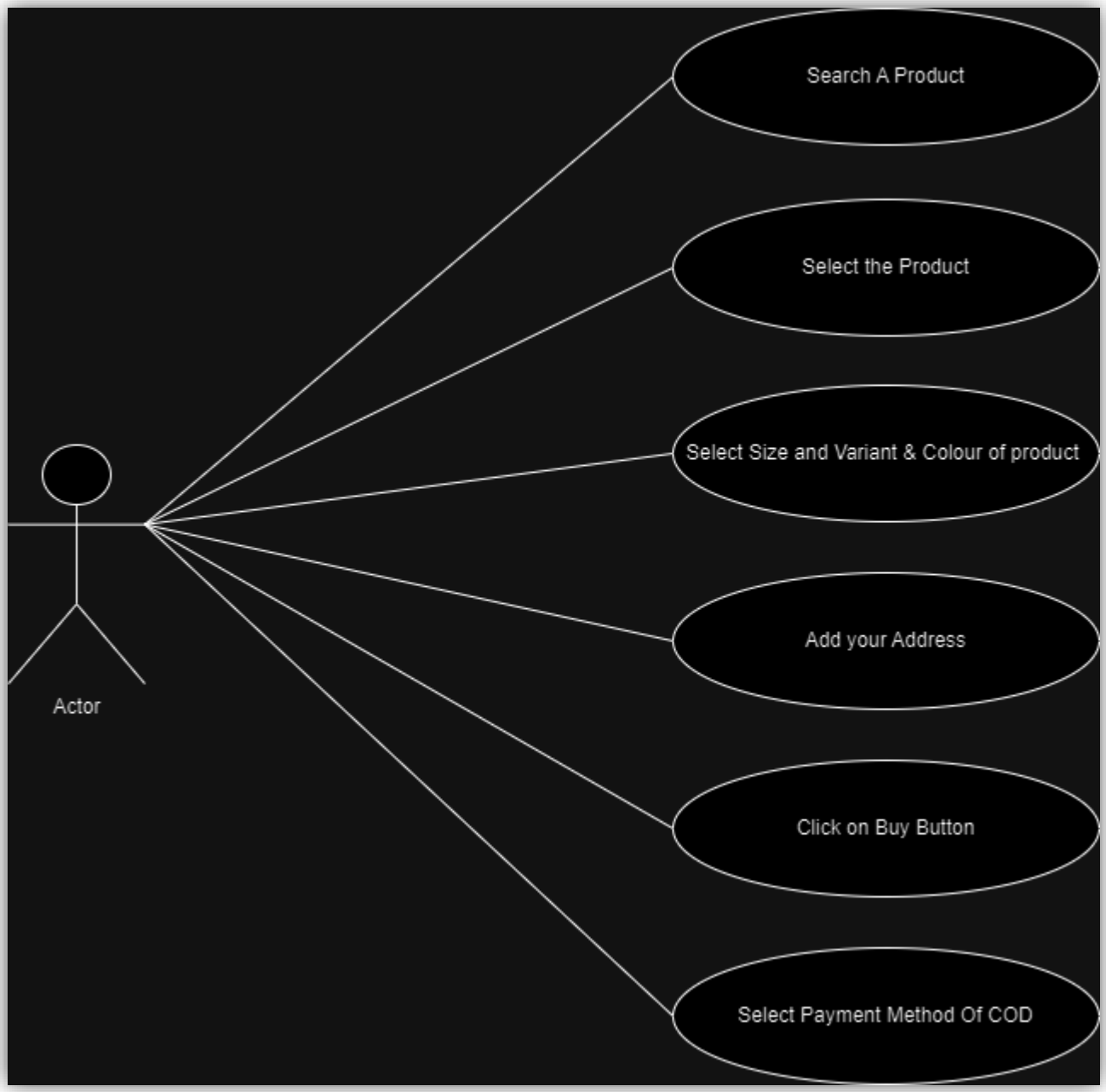
Software Requirements Specification (SRS) is a Complete Description Of the System to be Developed

A Software Requirements Specification (SRS) document is a comprehensive description of a software system to be developed. It outlines the system's functionality, performance, and constraints, serving as a guide for developers, testers, and stakeholders throughout the software development lifecycle.

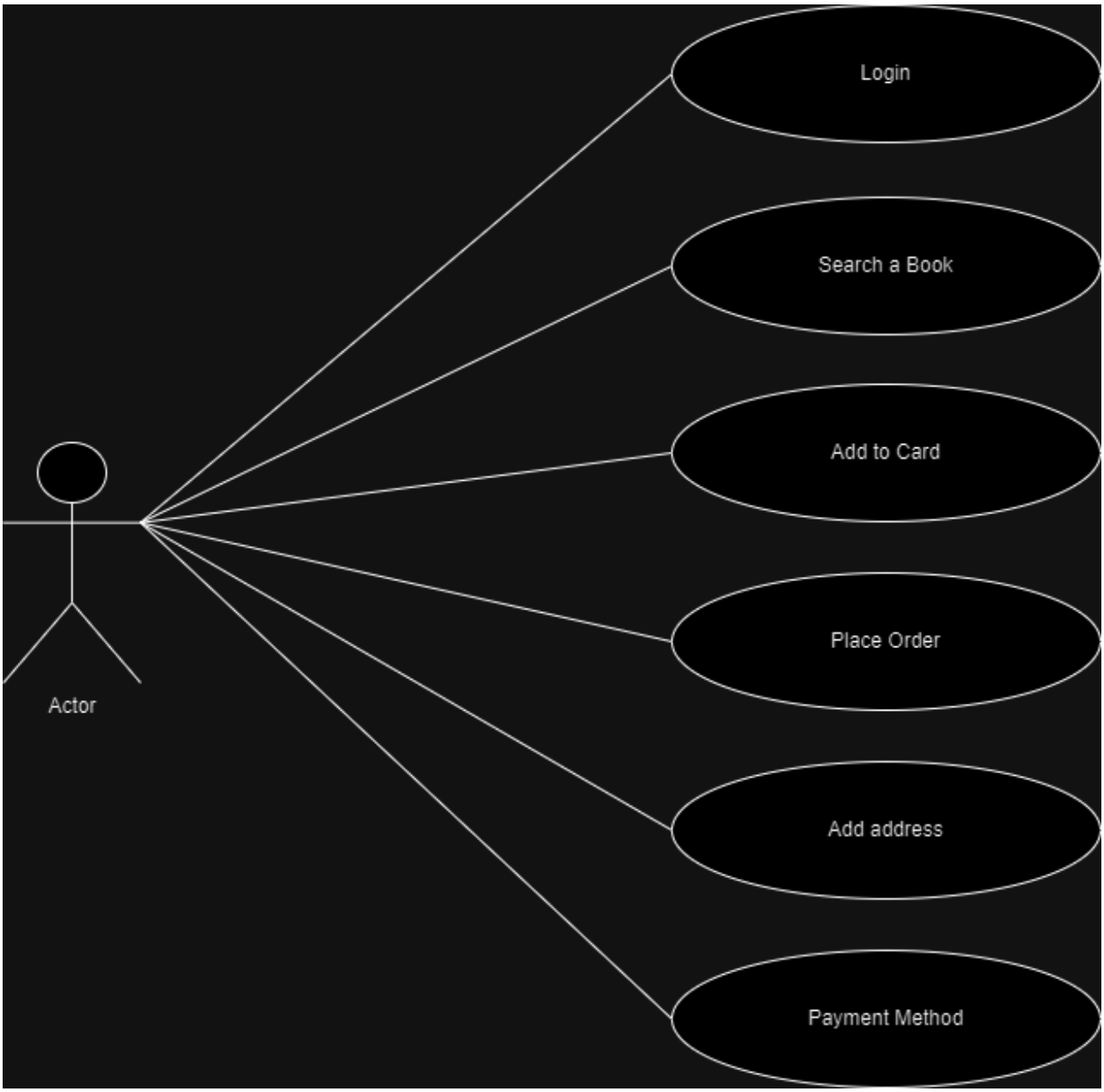
Types of Requirements

- Customer Requirements
- Functional Requirements
- Non-Functional Requirement

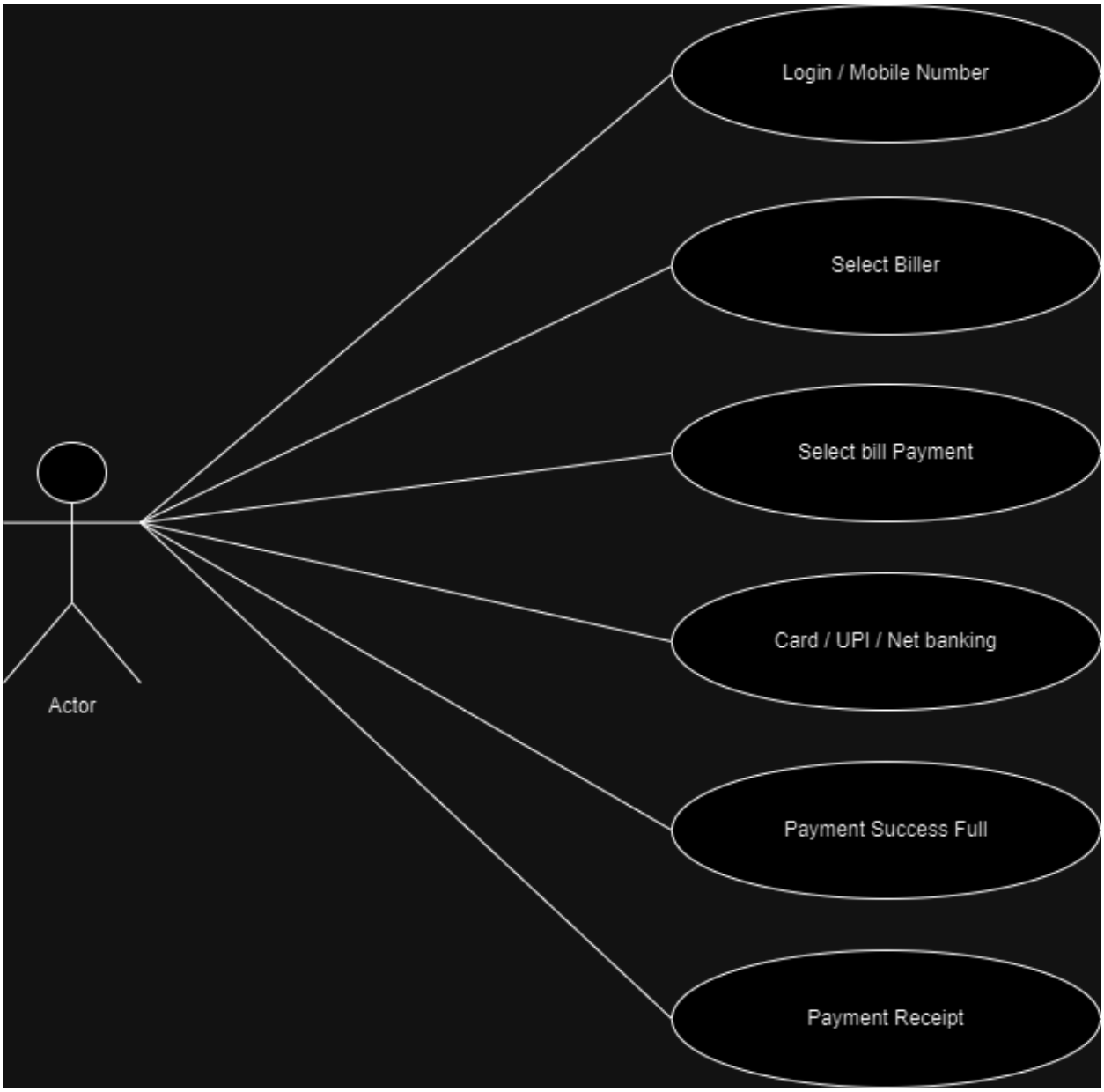
5. Draw use case on Online shopping product using COD.



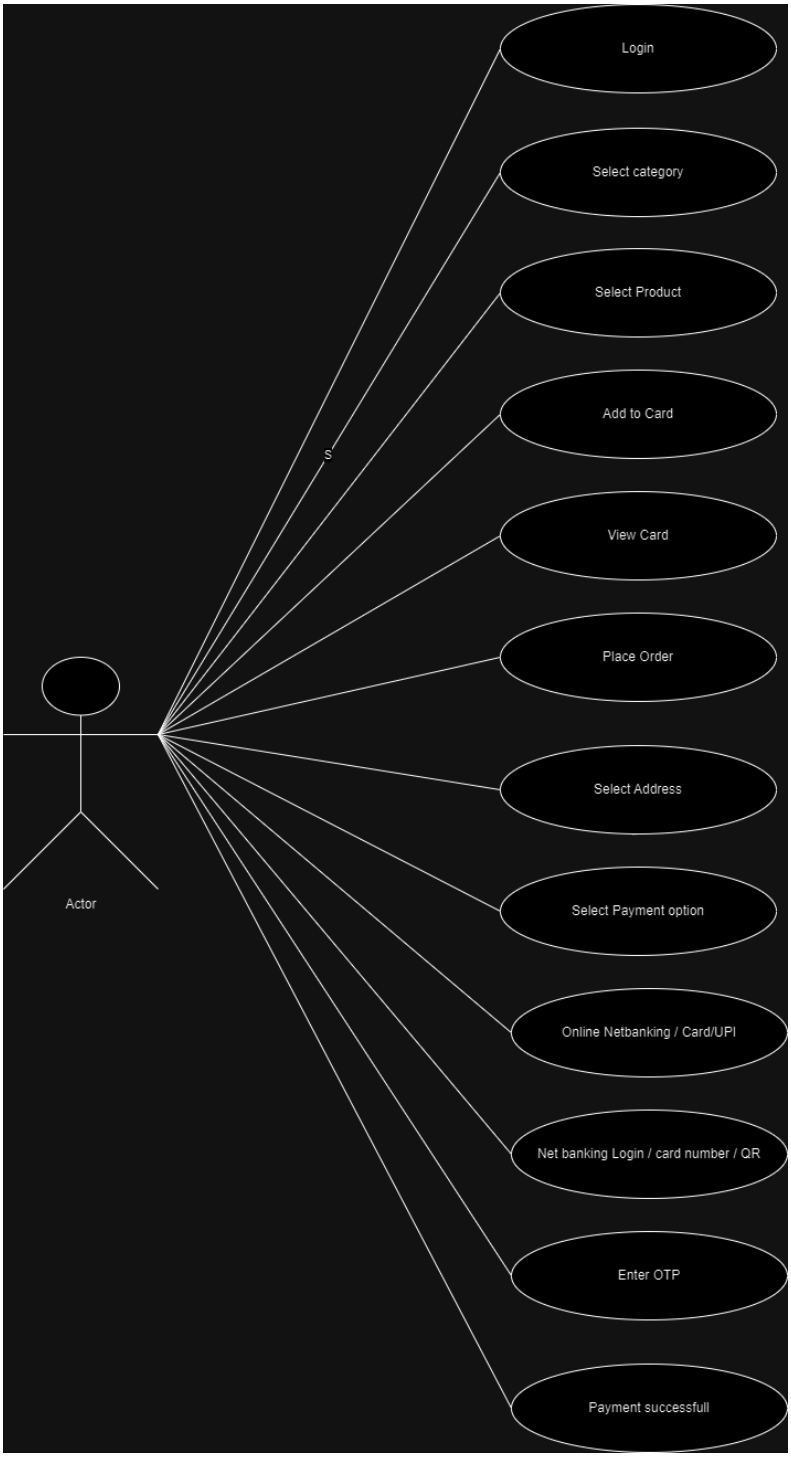
6. Draw use case on Online Book Shopping



7.draw Use case on Online Bill Payment System (Paytm)



8.Draw use case on Online Shopping Product using Payment gateway



9. What is oops? (Object Oriented Programming)

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic.

It's a programming paradigm that uses "objects" to design and implement software. The key concepts of OOP include

But actually, programming is not so easy, because a real good program is not easily programmed. It needs the programmers' lots of wisdom, lots of knowledge about programming and lots of experience.

10.Write Basic Concepts of oops

- basic concept of oops-
- Class
- Object
- Encapsulation
- Polymorphism
- Abstraction

11. What is Object?

- Tangible Things as a car, printer, ...
- Roles as employee, boss, ...
- Incidents as flight, overflow, ...
- Interactions as contract, sale, ...
- Specifications as color, shape, ...

basic unit for OOP. object will give the memory to the class. object will represent the relevant class.

12. What is a Class?

Class is a collection of a data member (variables) and member function with its behavior.

Class is a blueprint or a template to describe the properties and behavior of the objects.

13. What is encapsulation?

In object-oriented programming (OOP), encapsulation is the practice of bundling data and methods that work with that data into a single unit. This can be thought of as similar to a medicine capsule that can't be seen from the outside, where the inner workings are hidden and only what's necessary is exposed.

14. What is inheritance?

One class (Super, Base) inherits the properties of another class (Sub, Derived).

Types of Inheritance:

Single Inheritance

---

15. What is polymorphism?

An ability to take one name having many different forms.

Compile time Polymorphism: (Operator Overloading)

Method name should be same in single class, but its behavior (Arguments & Data type) is

different.

Run time Polymorphism (Operator Overriding)

Method should be same in super class and sub class but its behavior is different.

16. What is agile methodology?

"Agile SDLC model is a combination of iterative and incremental process models with focus on

process adaptability and customer satisfaction by rapid delivery of working software product."

17. Explain Phases of the waterfall model.

"The waterfall model is a classical software lifecycle that models the software development as a

step-by-step "waterfall" between the various development phases."

1. Requirements Collection/Gathering

Three types of problems can arise

Lack of clarity, Requirement confusion, Requirement Amalgamation

2. Analysis

3. Design (Low Level Design & High Level Design)

4. Implementation / Coding

5. Testing

6. Maintenance

Corrective Maintenance, Adaptive Maintenance, Perfective Maintenance

18. Explain Phases of the waterfall model.

"The waterfall model is a classical software lifecycle that models the software development as a

step-by-step "waterfall" between the various development phases."

1. Requirements Collection/Gathering

Three types of problems can arise

Lack of clarity, Requirement confusion, Requirement Amalgamation

2. Analysis

3. Design (Low Level Design & High Level Design)

4. Implementation / Coding

5. Testing

6. Maintenance

Corrective Maintenance, Adaptive Maintenance, Perfective Maintenance

19. Write phases of spiral model.

Phases of spiral modal –

1. Planning – determination of objectives, alternative and constraint.
2. Risk Analysis/Design - analysis of alternative and identification/resolution of risk.
3. Engineering/codding – development of the next level product.
4. Customer evaluation/Testing – assessment of the results of engineering.

20. Write agile manifesto principles.

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.

21. Explain working methodology of agile model and write pros and cons.

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product."

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.