1. What is SDCL?

G The software development lifecycle (SDLC) is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond.

There are many different SDLC models out there, but they all feature similar steps: planning, creating, testing, deploying, and monitoring. In the next section, we'll discuss these steps in more detail.

1. What is software testing?

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

3.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.

Agile Methods 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 areas like planning, requirements analysis,

design, coding, unit testing, and acceptance testing.

At the end of the iteration a working product is displayed to the

customer and important stakeholders.

4.What is SRS?

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill the needs of all stakeholders (business, users).

5. What is oops?

Identifying objects and assigning responsibilities to these objects. Objects communicate to other objects by sending messages. Messages are received by the methods of An object is like a black box.

⚫ The internal details are hidden. 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.

6. write basic concept of oops.

Object

Class

Encapsulation

Inheritance

Polymorphism

Overriding

Overloading

Abstraction

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. Annan "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.

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.

A class represents an abstraction of the object and abstracts the

properties and behaviour of that object.

Class can be considered as the blueprint or definition 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.

In the case of a car or laptop, there will be a blueprint or design created first

and then the actual car or laptop will be built based on that.

We do not actually buy these blueprints but the actual objects.

9. What is encapsulation?

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 work on that

data in the same place. While working with procedural languages, it is

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 (methods) of an object into a single unit;

and the unit here is a Class (or interface).

Encapsulate in plain English means to enclose or be enclosed in or as if

in a capsule. In Java, a class is the capsule (or unit).

10. What is inheritance?

Inheritance means that one class inherits the characteristics of

another class. This is also called a “is a” relationship

One of the most useful aspects of object-oriented programming is code

reusability. As the name suggests Inheritance is the process of forming

a new class from an existing class that is from the existing class called

as base class, new class is formed called as derived class.

This is a very important concept of object-oriented programming

since this feature helps to reduce the code size.

Inheritance describes the relationship between two classes. A class can

get some of its characteristics from a parent class and then add unique

features of its own.

In general, Java supports single-parent, multiple-children inheritance

and multilevel inheritance (Grandparent-> Parent -> Child) for classes

and interfaces. Java supports multiple inheritances (multiple parents,

singleton child) only through interfaces.

In a class context, inheritance is referred to as implementation

inheritance, and in an interface .

11. What is polymorphism?

Polymorphism means “having many 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).

The ability to use an operator or function in different ways in other words

giving different meaning or functions to the operators or functions is called

polymorphism.

Poly refers to many. That is a single function or an operator functioning in

many ways different upon the usage is called polymorphism.

E.g. the message display Details() of the Person class should give

different results when send to a Student object (e.g. the enrolment

number).

⚫ The ability to change form is known as polymorphism.

There is two types of polymorphism in Java

Compile time polymorphism(Overloading)

Runtime polymorphism(Overriding)

12. Draw use case on online book shopping.

USE CASE DIAGRAM

Ship order

Receive

Send

Register details

Provide Details

Provide New Details

Customer

Browse And Select

Update details

Bookshop staff

Create order

Provide Item Details

Update Items

15. Write SDLC phases with basic introduction.

PLANNING: obtain approval for project, Initiate, Assess feasibility, plan, schedule.

ANALYSIS: Understand business needs and processing needs

DESIGN: Define solution system based on requirement and analysis decision

IMPLEMENTATION: Construct, test, train users, install new system

MAINTENANCE: Keep system healthy and improve

16. Explain phases of the water fall model.

WATERFALL Model

Requirements

Analysis

Designing

Coding

Testing

Implementation

The waterfall is unrealistic for many reasons,

especially:

Requirements must be “frozen” to early in the life cycle

Requirements are validated too late.

17. Write phases of spiral model.

Phases of Spiral Model

This side depicts the phases of the spiral model, such as requirement analysis, design coding, testing & risk analysis, and the steps performed at each stage.

Requirement Analysis

• Collection of business requirements is the first step in the spiral model process

Documenting of requirement specification, unit requirements, and subsystem specifications will be included in the following spirals

• Add Text Here

Design

We will prepare the logical design, architectural design, flow charts, decision tree, and so on in stage 2 of the spiral model

• Add Text Here

• Add Text Here

Coding

• We will design the product following the client's specifications while also soliciting input from the client

In every cycle, this step pertains to creating the actual application

■ Add Text Here

Testing & Risk Analysis

We will test the build after the first cycle once the development is done successfully

We will also examine the risk of the program on several factors such as risk management, detection, and observation of technical feasibility

■ Add Text Here

Requirement Analysis

Design

Coding

Testing & Risk Analysis

This slide is 100% editable Adapt to your needs & capture your audience's attention.

18. Phases of Spiral Model

This side depicts the phases of the spiral model, such as requirament analysis, design coding, testing & risk analysis, and the steps performed at each stage.

Requirement Analysis

• Collection of business requirements is the first step in the spiral model process

Documenting of requirement specification, unit requirements, and subsystem specifications will be included in the following spirals

• Add Text Here

Design

We will prepare the logical design, architectural design, flow charts, decision tree, and so on in stage 2 of the spiral model

• Add Text Here

• Add Text Here

Coding

• We will design the product following the client's specifications while also soliciting input from the client

In every cycle, this step pertains to creating the actual application

■ Add Text Here

Testing & Risk Analysis

We will test the build after the first cycle once the development is done successfully

We will also examine the risk of the program on several factors such as risk management, detection, and observation of technical feasibility

■ Add Text Here

Requirement Analysis

Design

Coding

Testing & Risk Analysis

This slide is 100% editable Adapt to your needs & capture your audience's attention.

18. Write agile manifesto principal.

1. highest priority: satisfy the customer

2. even late change of requirements is welcomed

3. Frequent delivery of working software

4. daily work together

5. motivated individuals is given environment and support they need, and trust them to get the job done

6. conveying information: face-to-face conversation

7. primary measure of progress: working software

8. agile processes promote sustainable development, stakeholders 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 maximising the amount of work not done - is essential

11. self-organising teams best architectures, requirements, and designs

12. team regularly reflects on how to become more effective

19. Explain working methodology of agile model and also pros and cons.

Pros:-

Is a very realistic approach to software development Promotes teamwork and cross training. Functionality can be developed rapidly and demonstrated. Resource requirements are minimum. Suitable for fixed or changing requirements Delivers early partial working solutions. Good model for environments that change steadily. Minimal rules, documentation easily employed. Enables concurrent development and delivery within an overall planned context. Little or no planning required Easy to manage Gives flexibility to developers.

Cons:-

Not suitable for handling complex dependencies.

More risk of sustainability, maintainability and extensibility.

An overall plan, an agile leader and agile PM practice is a must without

which it will not work.

Strict delivery management dictates the scope, functionality to be

delivered, and adjustments to meet the deadlines.

Depends heavily on customer interaction, so if customer is not

clear, team can be driven in the wrong direction.

There is very high individual dependency, since there is minimum

documentation generated.

Transfer of technology to new team members may be quite challenging

due to lack of documentation.