**MODULE – 1 FUNDEMENTAL**

* What is SDLC?

SDLC is a structure imposed on the development of a software product that defines the process for planning, implementation, testing, documentation and ongoing maintenance.

* What is Agile Methodology?

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

* What is SRS?

A Software Requirement Specifications (SRS) is a document that captures complete description about how the system is expected to perform. It is usually signed off at the end of requirements engineering phase.

It includes a set of use cases that describe all the interactions that the users will have with the software.

* What is OOPS

Object-Oriented Programming (OOP) is a computer programming model that organizes software design around data, or object, rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior.

OOP focus on the objects that the developers want to manipulate rather than the logic required to manipulate them. This approach to programming is well-suited for programs that are large, complex, and actively updated or maintained.

* Write basic concept of OOPS

The basic concept of OOPS:

* Object
* Class
* Encapsulation
* Inheritance
* Polymorphism
* Overriding
* Overloading
* Abstraction
* What is Object

An object is a single instance of a class, which contains data and methods working on that data.

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 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.

Object = Data+Methods

Or

To say something differently

* What is Class

A class represents an abstraction of the object and the abstracts the properties and behaviour of the object.

Class can be considered as the blueprint or a definition or a template for ana object and describes the properties and behaviour of that object, but without any actual existence.

Classes are used to create and manage new objects and support inheritance – a key ingredient in Object-Oriented Programming and a mechanism a reusing code.

* What is Encapsulation

Encapsulation is the process 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.

Encapsulation describes the idea of bundling data and methods that work on that data within one unit, like a class in Java. This concept is also often used to hide the internal representation, or state of an object from the outside. This is called information hiding.

* What is Inheritance

Inheritance is the procedure in which one class inherits the attributes and methods of another class. The class whose properties and of another class. The class whose properties and methods are inherited is known as the Parent class. And the class that inherits the properties from the parent class is the child class.

This is also called a “is a” relationship.

In a class context, inheritance is referred to as implementation inheritance, and in an interface context, it is also referred to as interface inheritance.

* What is polymorphism

Polymorphism is the ability of any data to be processed in more than one form. It allows different objects to respond to the same message in different ways, the response specific to the type of the object.

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.

Two types of polymorphism in Java

* Compile time polymorphism (overloading)
* Runtime polymorphism (overriding)
* What is RDBMS

The software used to store, manage, query and retrieve data stored in a relational database is called Relational Database Management System (RDBMS). The RDBMS provides an interface between users and applications and the database, as well as administrative functions for managing data storage, access and performance.

It provides a dependable method of storing and retrieving large amounts of data while offering a combination of system performance and ease of implementation.

* What is SQL

SQL is a Structured Query Language, which is a computer language for storing, manipulating, and retrieving data stored n relational database.

It is a standard language for Relational Database System. It enables a user to create, read, update, and delete relational databases and tables.

* Write SQL Commands

There are four types of SQL Commands

1. DDL - Data Definition Language
2. DQL - Data Query Language
3. DML – Data Manipulation Language
4. DCL – Data Control Language

DDL – Data Definition Language

|  |  |
| --- | --- |
| **Command** | **Description** |
| CREATE | Create a new table, a view of a table or other object in database. |
| ALTER | Modifies an existing database object, such as table. |
| DROP | Deletes an entire table, a view of a table or other object in the database. |

DQL – Data Query Language

|  |  |
| --- | --- |
| **Command** | **Description** |
| SELECT | Retrieves certain records from one or more tables |

DML – Data Manipulation Language

|  |  |
| --- | --- |
| **Command** | **Description** |
| Insert | Creates a record |
| Update | Modifies Records |
| Delete | Delete Records |

DCL – Data Control Language

|  |  |
| --- | --- |
| **Command** | **Description** |
| GRANT | Gives privilege to user |
| REVOKE | Takes back privileges granted from user |

* Draw use case of online book shopping

Register

Check Out

Order Books

customer

Log In

Sell Used Books

Review Books

* Draw use case on online bill payment system (paytm)

Merchant Credit Card Processing System

User

|  |
| --- |
| If approved  If not approved |

* Write SDLC phases with basic introduction

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.

The SDLC phases are

* Requirement Collection/Gathering
* Analysis
* Design
* Implementation
* Testing
* Maintenance

**SDLC Phases**

|  |  |
| --- | --- |
| Requirement 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 |

* Requirement Gathering

During this phase, all the relevant information is collected from the customer to develop a product as per their expectation.

* Analysis

The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished.

This phase defines the problem that the customer is trying to solve.

* Design

In design phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived.

* Implementation

Implementation/coding starts once the developer gets the Design document. The software design is translated into source code. All the components of the software are implemented in this phase.

* Testing

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

* Maintenance

Software maintenance is one of the activities in software engineering, and is the process of enhancing and optimizing deployed software, as well as fixing defects. The maintenance phase is the phase which comes after deployment of the software into the field.

* Explain the phases of the waterfall model

Requirements

Analysis

Design

Coding/

Implementation

Testing

Operation/

Deployment

Maintenance

* Requirements

Specifically, we need to know and understand what we must design, what we have to develop, its process, what functionality, etc. It provides input material to the product being made; thus, the upcoming product is studied, finalized and marked.

* Analysis

It results in designing models, schema, and business rules. It is divided into two parts:

* Requirement gathering and analysis: First, all the information and requirement for the development is gathered from the customer and processed for analysis. The primary role of the part is to eradicate incompleteness and inconsistencies related to software product development.
* Requirement Specification: The above-analyzed requirements are documented in an SRS document. It serves as a path between the customer and the SRS development team.
* Design

It is the next most important phase to be studied as it is used for system design. It helps in specifying in specifying software and hardware requirements for the product design. It also helps in the overall architecture of the system design.

* Implementation

In this phase, the input from the system design is taken, and it is first developed in small programs known as units, which are tested and implemented in the upcoming phase. Each unit in the implementation phase undergoes development, and its full functionality is tested, also known as unit testing.

* Testing

Testing is done to maintain the stability and feasibility of the software so that the client does not face any disturbances or bugs during its production. So, in this phase, the whole system is tested thoroughly for any faults and failures after implementation.

System testing consists of three different types of activities:

* Alpha Testing
* Beta Testing
* Acceptance Testing
* Operation/Deployment

Once the testing is done, the software is deployed to the user or customer system or released to the market

The deployment phase includes installation, migration and support of the complete system to the user or customer environment.

* Maintenance

This step comes just after installation, and it includes making the appropriate modification to the product or system or enhancing changes, or modifying attributes related to performance issues related to the system.

* Write the phases of spiral model

The four phases of spiral model are

* Requirement Analysis
* Design
* Coding
* Testing and Risk Analysis
* Write Agile Manifesto Principles
* **Individuals and Interactions** – in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
* **Working software** – Demo working software is considered the best means of communication.
* **Customer collaboration** – As the requirements cannot be gathered completely in the beginning of the project due to various factors customer collaboration is important to get proper product requirements.
* **Responding to change** – agile development is focused on quick responses to change and continuous development.
* What is Join?

A Join is a clause is used to combine rows from two or more tables, based on a related column between them.

* Write types of joints
* INNER JOIN - returns rows when there is a match in both tables.
* LEFT JOIN - returns all rows from left table, even if there are no matches in the right table.
* RIGHT JOIN - returns all rows from the right table, even if there are no matches in the left table.
* FULL JOIN – returns rows when there is a match in one of the tables.
* Explain working methodology of agile model and also write pros and cons

Req Analysis

Testing

Planning

Designing

Building

2-3 months

Planning

Designing

Building

Testing

Req Analysis

2-3 months

Planning

Designing

Building

Testing

Req Analysis

**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.
* Good models 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 and extensibility.
* An overall plan, ana 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 wrong direction.
* There is very high individual dependency.
* Transfer of technology to new team members may be quite challenging due to lack of documentation.
* Draw use case on online shopping product using COD

|  |
| --- |
|  |

* Draw usecase on online shopping product using payment gateway

USER

Add Item

View Cart

Proceed to pay

Select Delivery Address

Select Payment Address

Select Payment Method

Credit Card

Enter Details

Bank Authentication

Enter One Time Password

Confirm

View Orders