Unit 1

SDLC: https://www.javatpoint.com/software-engineering-software-development-life-cycle

Architectural design(High level design):

1. The software needs the architectural design to represents the design of software.
2. IEEE defines architectural design as "the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system."

HLD stands for **High-Level Design,** where the designer will only focus on the various models, like:

- **Decision Tables**
- **Decision Trees**
- **Flow Diagrams**
- **Flow Charts**
- **Data Dictionary**

The solution architect develops the High-level design, which is used to specifies the complete description or architecture of the application.

The HLD involves **system architecture, database design, a brief description of systems, services, platforms, and relationships** among modules.

HLD is also known as **macro-level or system design**. It changes the business or client requirement into **a High-Level Solution**.

The High-level design is created before the Low-Level Design.

Detailed design(Low level design):

- The high-level design fixes the what, and the low-level design is all about hows.

- The LLD has exhaustive application detailing that concentrates on how the different parts of the unit will work together.
- It gets into the details of how the components and classes work, the various properties of the classes, the definitions of the database, and the interfaces.
- The **LLD** stands for **Low-Level Design,** in which the designer will focus on the components like a **User interface (UI)**.
- The Low-level design is created by the **developer manager and designers**.
- It is also known as **micro-level or detailed design**. The LLD can change the **High-Level Solution** into a **detailed solution**.
- The Low-level design specifics the detailed description of all modules, which implies that the LLD involves all the system component's actual logic. It goes deep into each module's specification.

  HLD vs LLD: https://www.javatpoint.com/hld-vs-lld

Modularization

Modularization is a technique to divide a software system into multiple discrete and independent modules, which are expected to be capable of carrying out task(s) independently. These modules may work as basic constructs for the entire software. Designers tend to design modules such that they can be executed and/or compiled separately and independently.

Modular design unintentionally follows the rules of 'divide and conquer' problem-solving strategy this is because there are many other benefits attached with the modular design of a software.

Advantage of modularization:

- Smaller components are easier to maintain
- Program can be divided based on functional aspects
- Desired level of abstraction can be brought in the program
- Components with high cohesion can be re-used again
- Concurrent execution can be made possible
- Desired from security aspect

Structure Chart and Flow chart:

https://www.geeksforgeeks.org/difference-between-structure-chart-and-flow-chart/

Pseudo Code

https://www.codingninjas.com/codestudio/library/pseudocodes-in-software-engineering

Design principles/ Design for quality attributes

https://www.javatpoint.com/software-engineering-software-design-principles

https://betterprogramming.pub/10-design-principles-in-software-engineering-f88647cf5a07

Design Strategies: Function Oriented Design, Object Oriented Design.

https://www.geeksforgeeks.org/difference-between-function-oriented-design-and-object-oriented-design/

UNIT 2

Architectural Styles: https://www.geeksforgeeks.org/software-engineering-architectural-design/

Architectural Patterns: https://www.geeksforgeeks.org/types-of-software-architecture-patterns/

Architectural Tradeoff analysis method (ATAM): https://www.geeksforgeeks.org/architecture-tradeoff-analysis-method-atam/

Service oriented architecture(SOA): https://aws.amazon.com/what-is/service-oriented-architecture/#:~:text=Service%2Doriented%20architecture%20(SOA),other%20across%20platforms%20and%20languages.

Architecture of mobile app: https://os-system.com/blog/mobile-app-architecture-how-to-design-it/

Architecture of Network systems: https://www.javatpoint.com/computer-network-architecture

Architecture of Embedded systems: https://www.geeksforgeeks.org/architecture-of-an-embedded-system-set-3/

UML DIAGRAMS: https://www.javatpoint.com/uml-diagrams