High Level Design Vs Low Level Design

High Level Design	Low Level Design
Describes the main component that would be developed for resulting products	Describes the design of each element mentioned in HLD of the system
The system Architecture details, Database design, services and processes, the relation between various modules, and features.	Classes, Interfaces, relation between different classes, and actual logic of various component.

Monolithic Architecture

Monolithic architecture is a traditional software design model where all components of an application are integrated into a single, self-contained unit.This architecture style is characterized of a single code base which have all the elements such as Interface, Business logic and data access layer.

- **Single Codebase :** All application components are part of a single codebase and are compiled together into one executable or deployable unit.
- **Tightly coupled :** Components are tightly integrated, making it easier for them to communicate with each other. However, this tight coupling can also make the system more complex and harder to maintain as it grows.
- **Single Deployment :** The entire application is deployed as a single unit. Any changes or updates require the entire application to be rebuilt, tested, and redeployed.
- **Ease of Deployment :** For smaller applications, monolithic architecture can be easier to develop and manage because all parts of the application are in one place.
- **Performance :** In some cases, monolithic applications can offer better performance since all components are in the same process and can share memory.

However, there are also several challenges associated with monolithic architecture:

- **Scalability**: Scaling a monolithic application can be difficult. It typically involves running multiple instances of the same application, which may not be efficient if only one part of the application needs to scale.
- **Maintainability**: As the application grows, the codebase can become large and difficult to manage. Changes in one part of the application can inadvertently affect other parts.
- **Deployment**: Deploying a monolithic application can be complex because even small changes require redeploying the entire application.
- **Technology Stack:** It can be difficult to incorporate new technologies, as changes need to be made across the entire application.

Monolithic architecture can be a good choice for small to medium-sized applications or for projects where the application requirements are well-understood and unlikely to change significantly. For larger, more complex applications, or those requiring frequent updates and scalability, a microservices architecture might be more appropriate.

Distributed System Architecture

Distributed system architecture is a collection of individuals system connected through a network that share resources, communicate and coordinate to achieve common goals.

Advantages : Scalable, Low Latency, No Single point of failure

Disadvantages : Complex, Additional Management, Difficult to secure, Message may be lost between nodes.

Important Terms

Latency : It is the network and computation delay in response. Monolithic design will only have the Computational delay and distributed design will have both network and computational delay.

To reduce latency, we can do **caching**, use **CDN(content delivery network)** for static data.Caching is set on the main servers but the cdn are geographically distributed network of proxy servers.

Throughput : It is the volume of the work or data flowing through the system.It is measured in bits per second (amount of data transferred in the given time).

Distributed system have more throughput than monolithic systems.

Causes of Low Throughput : Increased Latency, Increase in incoming calls [Congestion]