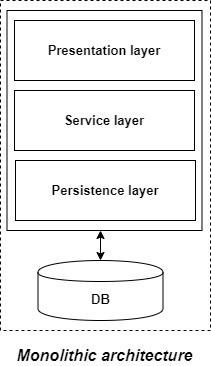
# Monolithic Vs Microservices architecture

**Monolithic applications**   
If all the functionalities of a project exist in a single codebase, then that application is known as a monolithic application. We all must have designed a monolithic application in our lives in which we were given a problem statement and were asked to design a system with various functionalities. We design our application in various layers like presentation, service, and persistence and then deploy that codebase as a single jar/war file. This is nothing but a monolithic application, where **“mono”**represents the single codebase containing all the required functionalities.



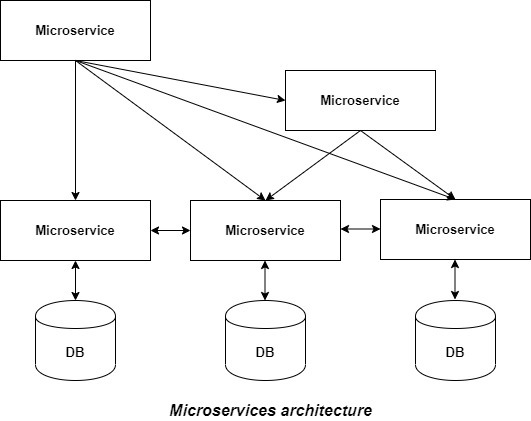
**Disadvantages of Monolithic applications:**

* It becomes too large with time and hence, difficult to manage.
* We need to redeploy the whole application, even for a small change.
* As the size of the application increases, its start-up and deployment time also increases.
* For any new developer joining the project, it is very difficult to understand the logic of a large Monolithic application even if his responsibility is related to a single functionality.
* Even if a single part of the application is facing a large load/traffic, we need to deploy the instances of the entire application in multiple servers. It is very inefficient and takes up more resources unnecessarily. Hence, horizontal scaling is not feasible in monolithic applications.
* It is very difficult to adopt any new technology which is well suited for a particular functionality as it affects the entire application, both in terms of time and cost.
* It is not very reliable, as a single bug in any module can bring down the entire monolithic application.

**Advantages of monolithic applications:**

* Simple to develop relative to microservices, where skilled developers are required in order to identify and develop the services.
* Easier to deploy as only a single jar/war file is deployed.
* Relatively easier and simple to develop in comparison to microservices architecture.
* The problems of network latency and security are relatively less in comparison to microservices architecture.
* Developers need not learn different applications, they can keep their focus on one application.

**Microservices**   
**This** is a Service Oriented Architecture. In the microservice architecture, there are a large number of **microservices**. By combining all the microservices, it constructs a big service. In the microservice architecture, all the services communicate with each other. Microservices are the small services that work together.



**Advantages of microservices:**

* It is easy to manage as it is relatively smaller.
* If there’s any update in one of the microservices, then we need to redeploy only that microservice.
* It is very easy for a new developer to onboard the project as he needs to understand only a particular microservice providing the functionality he will be working on and not the whole system.
* Each microservice can use different technology based on the business requirements.
* If a particular microservice goes down due to some bug, then it doesn’t affect other microservices and the whole system remains intact and continues providing other functionalities to the users.

**Disadvantages of microservices:**

* Being a distributed system, it is much more complex than monolithic applications. Its complexity increases with the increase in a number of microservices.
* Skilled developers are required to work with microservices architecture, which can identify the microservices and manage their inter-communications.
* Independent deployment of microservices is complicated.
* Microservices are costly in terms of network usage as they need to interact with each other and all these remote calls result in network latency.
* Microservices are less secure relative to monolithic applications due to the inter-services communication over the network.
* Debugging is difficult as the control flows over many microservices and to point out why and where exactly the error occurred is a difficult task.