Web Application : Application that runs on the internet. It is interactive in nature.

Website : it have the static content that doesn't gets updated.

Clients : System that initiates the communication over the network. A webserver can also act as a client while requesting for information/data.

Server : System that receives the request over the network.

What is required for this communication to happen ?

- Language independent
- Fast
- Enable communication over the network
- Light Weighted

The above is called **Rest**, it is a technique / style / standard to prove the above requirements.
Example : the client requests the backend server to give them the subscription plan of Netflix. Thus the client hits a URI of backend to get the data.

The above request can be of 4 types :
C.R.U.D : Create Read Update Delete

The request that is made to the backend to perform the CRUD task using HTTP verb[POST, GET, PUT, DELETE] is called a **REST** API { URI + HTTP verb }.

SOA [Service Oriented Architecture]

It means of breaking a bigger Application to more smaller granular level for better reusability.

Advantages

- **Selective Scaling :** We can select the most frequently requested API to be selected and scaled up for better latency and API throughput.
- **Different Tech Stacks**
- **Loose Coupling**
- **Agile**

Disadvantages

- **Higher latency**
- **Complex to secure**
- **Cascading failures**
- **Complex understanding**

Microservice Architecture

Microservices Architecture is an evolved version of SOA that promotes software components to be more loosely coupled.

It is most granulated type of architecture design and every service is completely independent of other.

SOA	Microservices Architecture
Service can share data storage	Each microservice have its own independent data storage.
Less Scalable	Highly Scalable
Deployment is time consuming	Deployment is easy and less time consuming
Focused on maximizes application service reusabaility	More Focusing on decoupling

Tier Architecture

A web application can be designed according to the n-tier architecture where tier are different layers of architecture.

A Tier is a logical separation between different components of the application.

- **1 Tier** : A single machine have all three components, i.e, Frontend, Backend and Database.
- **2 Tier** : Two machine have all the three components. One machine have Frontend and another will have Backend and Database.
- **3 Tier** : All the component are at different machine. One machine have Frontend and another have Backend and third will have Database.