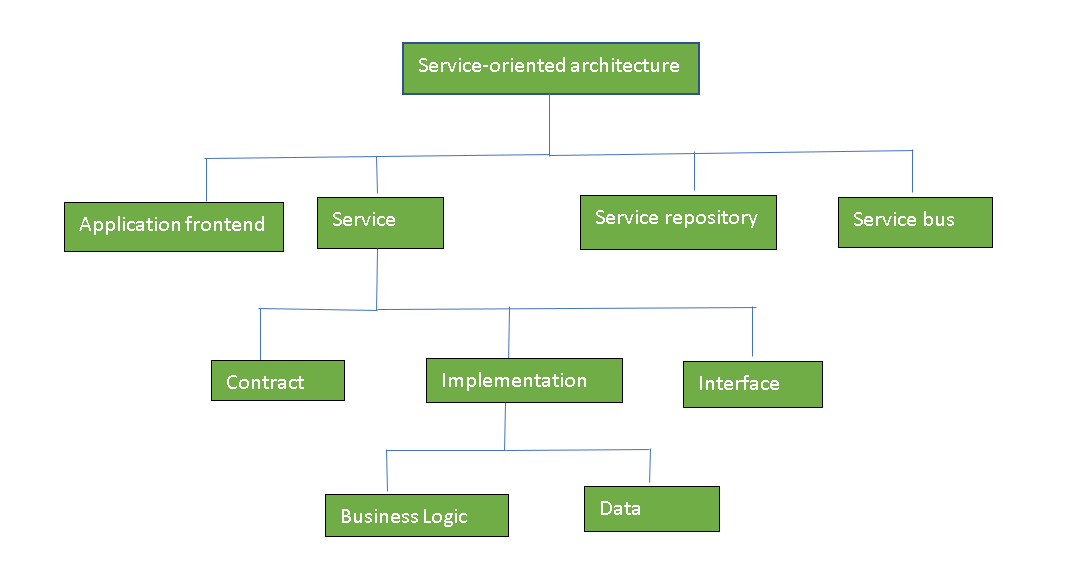
**Service-Oriented Architecture (SOA) – Simplified Notes**



* **Definition:**
* SOA stands for Service-Oriented Architecture.
* It is a way of designing software where services (small functional units) are used to build applications.
* These services are shared over a network like the Internet.
* 2. **Purpose of SOA:**
* To make software components reusable.
* To allow different systems to communicate with each other easily.
* To integrate multiple applications smoothly.
* 3. **Key Points**:

1. Each service performs a full business function (like payment, login, etc.).
2. Services are accessed via network calls (using protocols like HTTP, SOAP, REST).
3. Common standards are used so that services can be easily connected.
4. Developers can combine services to create new applications.
5. SOA is not the same as microservices (SOA is broader and more centralized).

* 4**. Features of SOA:**
* Reusable – Same service can be used in different applications.
* Loosely Coupled – Services work independently.
* Interoperable – Works across different platforms or languages.
* Discoverable – Services can be searched and found when needed.
* Scalable – Easy to add more services or handle more users.
* **Benefits:**
* Saves time and cost in development.
* Promotes easy integration between old and new systems.
* Improves flexibility and maintenance.
* Helps in building large, enterprise-level applications.
* **Use Case Examples:**
* Banking apps using a payment gateway service.
* Travel websites using a hotel booking API.
* Government systems sharing services like identity verification.
* **Disadvantages of Service-Oriented Architecture (SOA)**
* **High Overhead:**
  + Every time services talk to each other, they check the input values.
  + This slows down the system and increases load and response time.
* **Expensive Setup:**
  + Building an SOA system needs a lot of money in the beginning.
  + Investment is required for special tools, servers, and developers.
* **Complex Service Management:**
  + Services send messages to each other to complete tasks.
  + These messages can be in huge numbers (even millions).
  + Managing and tracking all these messages becomes difficult and confusing.
* **Security Issues:**
  + Since services are open over a network, they are more exposed to security risks.
* **Slower Performance:**

SOA uses XML or web service protocols which are slower than direct code execution.

* **Versioning Problems:**
  + Updating one service may affect others if not handled properly.

**Advantages of Service-Oriented Architecture (SOA)**

1. **Service Reusability:**
   * You can use the same service in different applications.
   * No need to build everything from scratch each time.
2. **Easy to Maintain:**
   * Each service works separately.
   * If you need to change one service, it won’t affect others.
3. **Works on Any Platform:**
   * You can combine services from different systems (like Windows, Linux, etc.).
   * They still work together easily.
4. **Easily Available:**
   * Services can be accessed quickly when needed.
   * They are shared over the network and available on request.
5. **More Reliable:**
   * If something goes wrong, it's easier to fix a small service than a large program.

**6.Scalable:**

* + Services can run on many servers.
  + It helps the system grow easily and handle more users.