1. What are the main benefits of SOA?

* Services are reusable, if the same service is already created we can alter that according to our requirements and can be reused.
* Services are easily maintained as updating, upgrading and maintaining the services.
* SOA services are reliable and easy form testing, debugging and any other maintenance.
* SOA are platform independent can be implemented on any system that needs them.

1. How can you achieve loose coupling in SOA

* Loose is nothing but, the bonding between the service ina system are coupled loosely, so that if one service is effected that doesn.t affect the other services that are coupled to that service.
* If one service gets affected due to overload or some other issue, the issue can be solved in that particular service without affecting the functionality of other services.

1. Are web services and SOA the same?

* SOA is an architectural concept which has many services that are loosely coupled to perform a bigger service where as Web services are basic building block that comes under SOA , when multiple web services are combined we get an application that comes under the SOA which provides the service.

1. What is a reusable service?

* Reusable service is one of the most important and advantageous benefit of the SOA .
* Services are re-used many times in different applications irrespective of the other interacting components in that system
* Reusable service are cost efficient and saves lot time.

1. What are the disadvantages of SOA?

* High Bandwidth server: it involves a high-speed server with plenty of information measure to run an internet service.
* Extra Overload: In SOA, all inputs square measures its validity before it’s sent to the service due to this extra load will be on the server and issue might occur.
* High Cost : It is expensive in terms of human resource, development, and technology.

1. What is ESB and where does it fit in?

* The Enterprise Service Bus (ESB) is a software architecture which connects all the services together over a bus like infrastructure.
* ESB provides access of all the services to the end users and connects with all the services and applications which is further connected to database.

1. In SOA do we need to build a system from scratch?

* No, if we need to integrate any existing system you just can loosely couple wrappers which help in wrapping all customer services and expose all functionalities in a generic manner.

1. What is the most important skill needed to adopt SOA ?technical or cultural?

* Cultural skill is the most important skill that needs to be developed to adopt SOA because cultural skill helps to think more about service oriented which provide better service to customer and business will be improved rather technology oriented.

1. List down the advantages of Microservices Architecture?

* Software built as microservices can be broken down into multiple component services which makes deployment and redeployment of the service can be done independently.
* Better fault isolation; if one microservice fails, the others will continue to work
* Code for different services can be written in different languages.
* The microservice architecture enables continuous delivery.
* Scalability and reusability, as well as efficiency.
* Components can be spread across multiple servers or even multiple data centres

1. What are the best practices to design Microservices?
2. Use distributed configuration
3. Keep code at a similar level of maturity
4. Do a separate build for each microservice
5. Deploy in containers
6. How does Microservice Architecture work?

Microservices architecture is nothing but further breaking down of services in to micro level and all the services are connected to the API gateway. When a costumer comes and gives his requirement at the API gateway in the form of URL the request is guided to the particular Microservice with help of rest api. Rest API takes all the request from the end user and works based on his requirement.

1. What are the pros and cons of Microservice Architecture

Pros:

1. We can develop and deploy each microservice on different programming language and developer tools.
2. Microservices use API’s and communication protocols to interact with each other
3. The teams can develop, deploy and maintain the microservices independently.
4. It is easier to maintain and debug a microservice than a complex application.

Cons:

1. need to introduce Devops tools such as CI/CD servers, configuration management platforms
2. Having to maintain a network lead to other kinds of issues
3. Needs more collaboration between teams
4. What is the difference between Monolithic, SOA and Microservices Architecture?

SOA:

Service-oriented architecture is a style of software design where services are provided to the other components by application components, through a communication protocol over a network

### Monolithic:

### Monolith means composed all in one piece. The **Monolithic** application describes a single-tiered **software** application in which different components combined into a single program from a single platform. This shares a single database for the whole application.

### Microservices:

### Microservices are an approach to application development in which a large application is built as a suite of modular services. Each module supports a specific business goal and uses a simple, well defined interface to communicate with other sets of services. Instead of sharing a single database as in Monolithic application, each microservice has its own database.

### 

1. What are the challenges you face while working Microservice Architectures?
2. Managing: As the no of microservices increase managing them is really abig task.
3. Monitoring: When a problem occurs its hard to find the path if ther is no good monitoring practise.
4. Devops culture: Lack of knowledge on the devops shows impact on the delivery time and response to the business needs
5. Testing: Testing is more complex in microservice environment due to different services.
6. What are the characteristics of Microservices?
7. Microservice architecture style is an approach for developing a single application with a group of small services.
8. Services are built based on the business requirements.
9. Each service should have separate database layer.
10. Each service can have independent codebase, CI/CD tooling sets.
11. Each service can be tested in isolation without dependent on other services.
12. Each service can be programmed using different programming language and developer tools.