Monolithic and Microservice

Monolith architecture -

It is a large system that runs on one or multiple machines, where all clients are connected to the same machine. The advantages of this architecture include ease of deployment, faster performance due to the absence of network calls, and simplified testing. However, it can be difficult for new team members to understand the entire system and deployments may become complicated as the codebase grows.

Advantages:

1. Easy to manage with a small and cohesive team
2. Less moving parts, easier deployments
3. Avoids duplication of code and faster as all logic is in one box
4. This is faster, we are not making any network (remote procedure call) its all in the box , local calls.
   Disadvantages:
5. Requires a lot of context for new team members
6. Complicated deployments(for new change in the code) and frequent monitoring
7. Single point of failure.
8. Tests are more complicated
9. Too much responsibility on each server, causing system collapse if one fails

Disadvantages:

1. Inflexibility: Monolithic architecture is rigid and does not allow for changes to be made to individual components without affecting the entire system.
2. Lack of scalability: Monolithic systems are not scalable and struggle to handle large amounts of traffic or data.
3. Difficulty in deployment and testing: Monolithic architecture makes it difficult to deploy and test individual components, as they are tightly integrated with the rest of the system.

Microservices architecture -

They are individual business units that have all the data and functions relevant to a specific service. They talk to their own dedicated databases and may communicate with a gateway that connects to the clients. The benefits of microservices include easier scalability, parallel development, and the ability for new team members to focus on a specific service. However, there can be a large number of moving parts and it can be challenging to manage and maintain the individual services.

Advantages:

1. Easier to scale
2. Assigning tasks based on service, reducing context for new team members
3. Facilitates parallel development
4. Reduced tight coupling between developers and services
   Disadvantages:
5. They are not easy to design.
6. Complexity in managing multiple services and intercommunication
7. Increased need for monitoring and coordination between services
8. Increased complexity in testing.

Disadvantages:

1. Complexity: Microservices architecture can be complex and difficult to manage, especially as the number of services increases.
2. Inter-service communication: Inter-service communication and coordination can become challenging as the number of services increases.
3. Monitoring and debugging: Monitoring and debugging microservices can be difficult, as there are many moving parts and the interactions between services can be complex.

The choice between monolith and microservice depends on the size and nature of the team, the system requirements, and the scalability goals.