**Microservices** is a service-oriented architecture pattern wherein applications are built as a collection of various smallest independent service units. It is a [software engineering](https://www.guru99.com/what-is-software-engineering.html) approach that focuses on decomposing an application into single-function modules with well-defined interfaces. These modules can be independently deployed and operated by small teams that own the entire lifecycle of the service.

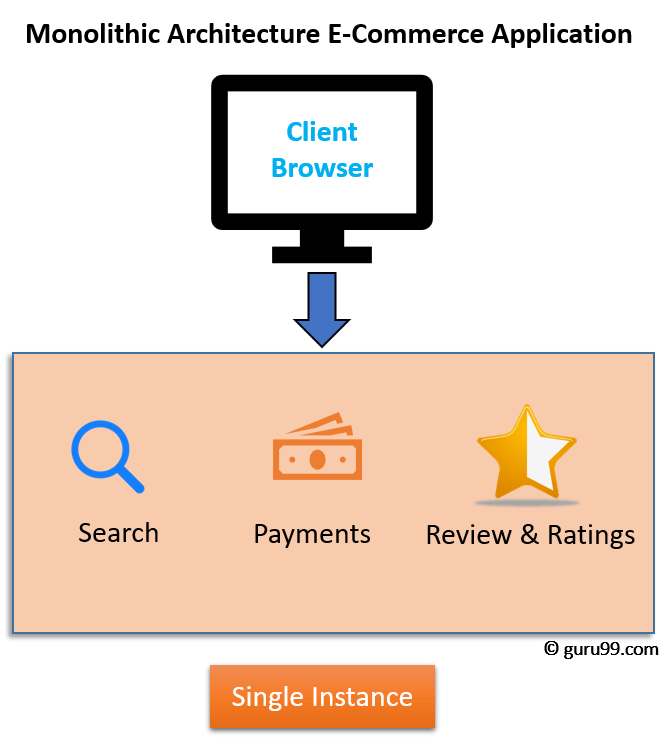
The term “micro” refers to the sizing of a microservice which must be manageable by a single development team ( 5 to 10 developers). In this methodology, big applications are divided into smallest independent units.

In this beginners Microservices tutorial, you will learn-

**What is Monolithic Architecture?**

In layman terms, you can say that Monolithic architecture is like a big container in which all the software components of an application are clubbed into a single package.

Let’s discuss an example of an eCommerce store in context of a Monolithic architecture.



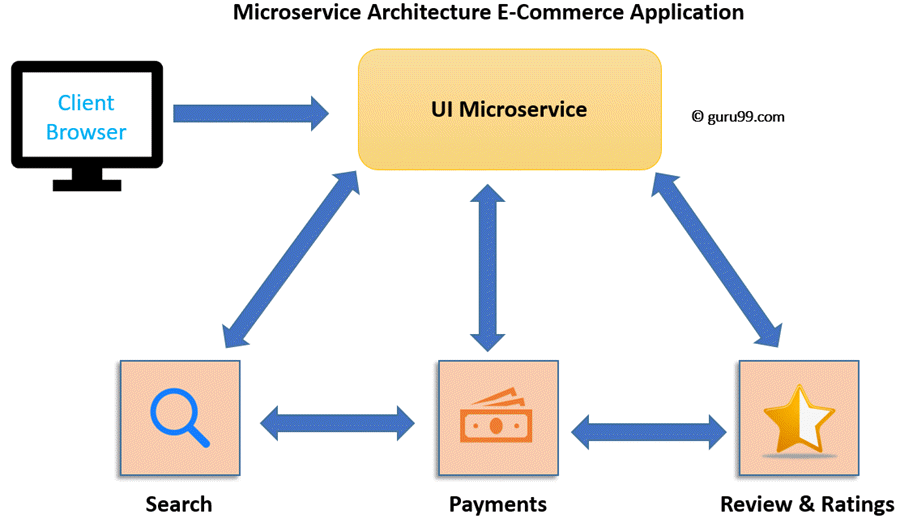
Monolithic Architecture of eCommerce Application

In any e-commerce application, there are some standard features like Search, Review & Ratings, and Payments. These features are accessible to customers using their browser or apps. When the developer of the eCommerce site deploys the application, it is a single Monolithic unit. The code for different features like Search, Review & Ratings, and Payments are on the same server. To scale the application, you need to run multiple instances(servers) of these applications.

**What is Microservice Architecture?**

**Microservice Architecture** is an architectural development style that allows building applications as a collection of small autonomous services developed for a business domain. It is a variant of structural style architecture that helps arrange applications as a loosely coupled service collection. The Microservice Architecture contains fine-grained services and lightweight protocols.

Let’s take an example of e-commerce application developed with microservice architecture. In this Microservices architecture example, each microservice is focused on single business capability. Search, Rating & Review and Payment each have their instance (server) and communicate with each other.



Microservices Architecture

## Microservices vs. Monolithic Architecture

| **Microservices** | **Monolithic Architecture** |
| --- | --- |
| Every unit of the entire application should be the smallest, and it should be able to deliver one specific business goal. | A single code base for all business goals |
| Service Startup is relatively quick | Service startup takes more time |
| Fault isolation is easy. Even if one service goes down, other can continue to function. | Fault isolation is difficult.  If any specific feature is not working, the complete system goes down. In order to handle this issue, the application needs to re-built, re-tested and also re-deployed. |
| All microservices should be loosely coupled so that changes made in one does not affect the other. | Monolithic architecture is tightly coupled. Changes in one module of code affect the other |
| Businesses can deploy more resources to services that are generating higher ROI | Since services are not isolated, individual resource allocation not possible |
| More hardware resources could be allocated to the service that is frequently used. In the e-commerce example above, more number of users check the product listing and search compared to payments. So, more resources could be allocated to the search and product listing microservice. | Application scaling is challenging as well as wasteful. |
| Microservices always remains consistent and continuously available. | Development tools get overburdened as the process needs to start from the scratch. |
| Data is federated. This allows individual Microservice to adopt a data model best suited for its needs. | Data is centralized. |
| Small Focused Teams. Parallel and faster development | Large team and considerable team management effort is required |
| Change in the data model of one Microservice does not affect other Microservices. | Change in data model affects the entire database |
| Interacts with other microservices by using well-defined interfaces | Not applicable |
| Microservices work on the principle that focuses on products, not projects | Put emphasize on the entire project |
| No cross-dependencies between code bases. You can use different technologies for different Microservices. | One function or program depends on others. |

## Microservice Challenges

* MicroServices rely on each other, and they will have to communicate with each other.
* Compared to monolithic systems, there are more services to monitor which are developed using different [programming languages](https://www.guru99.com/best-programming-language.html).
* As it is a distributed system, it is an inherently complex model.
* Different services will have its separate mechanism, resulting in a large amount of memory for an unstructured data.
* Effective management and teamwork required to prevent cascading issues
* Reproducing a problem will be a difficult task when it’s gone in one version, and comes back in the latest version.
* Independent Deployment is complicated with Microservices.
* Microservice architecture brings plenty of operations overhead.
* It is difficult to manage application when new services are added to the system
* A wide array of skilled professionals is required to support heterogeneously distributed microservices
* Microservice is costly, as you need to maintain different server space for different business tasks.

## SOA vs. Microservices

SOA services are maintained in the organization by a registry which acts as a directory listing. Applications need to look up the services in the registry and invoke the service.

In another world, [SOA](https://www.guru99.com/soa-principles.html) is just like an orchestra where each artist is performing with his/her instrument while the music director gives instructions to all.

On the other end, Microservices is a form of service-oriented architecture style wherein applications are built as a collection of different smaller services instead of one software or application.

Microservices is just like a troupe where each dancer is independent and know what they need to do. So, if they miss some steps, they know how to get back on the correct sequence. Now in this Microservices architecture tutorial, let’s learn about the difference between SOA and Microservices.

Here is a detailed comparison between SOA and Microservices

| **Parameter** | **SOA** | **Microservices** |
| --- | --- | --- |
| Design type | In SOA, software components are exposed to the outer world for usage in the form of services. | Micro Service is a part of SOA. It is an implementation of SOA. |
| Dependency | Business units are dependent. | They are independent of each other. |
| Size of the Software | Software size is larger than any conventional software | The size of the Software is always small in Microservices |
| Technology Stack | The technology stack is lower compared to Microservice. | Microservice technology stack could be very large |
| Nature of the application | Monolithic in nature | Full stack in nature |
| Independent and Focus | SOA applications are built to perform multiple business tasks. | They are built to perform a single business task. |
| Deployment | The deployment process is time- consuming. | Deployment is straightforward and less time-consuming. |
| Cost – effectiveness | More cost-effective. | Less cost-effective. |
| Scalability | Less compared to Microservices. | Highly scalable. |
| Business logic | Business logic components are stored inside of single service domain Simple wire protocols(HTTP with XML JSON) API is driven with SDKs/Clients | Business logic can live across domains enterprise Service Bus like layers between services Middleware |

## Microservices Tools

### 1) Wiremock: Testing Microservices

WireMock is a flexible library for stubbing and mocking web services. It can configure the response returned by the HTTP API when it receives a specific request. It is also y used for testing Microservices.

**Download link:** [**http://wiremock.org/**](http://wiremock.org/)

### 2) Docker

Docker is open source project that allows us to create, deploy, and run applications by using containers. By using these containers, developers can run an application as a single package. It allows you to ship libraries and other dependencies in one package.

**Download link:** [**https://www.docker.com/**](https://www.docker.com/)

### 3) Hystrix

Hystrix is a fault tolerance java library. This tool is designed to separate points of access to remote services, systems, and 3rd-party libraries in a distributed environment like Microservices. It improves overall system by isolating the failing services and preventing the cascading effect of failures.

**Download Link:** <https://github.com/Netflix/Hystrix>

## Best Practices of Microservices Architecture

* Separate data store for each Microservice
* Keep code of a similar level of maturity.
* Separate build for each Micro service.
* Always treat – severe as stateless.

## Summary

* Microservices is a service-oriented architecture pattern wherein applications are built as a collection of various smallest independent service units.
* Microservice Architecture is an architectural development style that allows building an application as a collection of small autonomous services developed for a business domain.
* Monolithic architecture is like a big container in which all the software components of an application are clubbed into a single package
* In a Microservice, every unit of the entire application should be the smallest, and it should be able to deliver one specific business goal
* In Monolithic architecture, large code base can slow down the entire development process. New releases can take months. Code maintenance is difficult
* Two types of Microservices are 1) Stateless 2) Stateful
* Microservices in Java rely on each other, and they will have to communicate with each other. Helps you to give emphasizes on a specific feature and business needs
* Service-oriented architecture shortly known as SOA is an evolution of distributed computing based on the request or reply design model for synchronous and asynchronous applications
* In SOA, software components are exposed to the outer world for usage in the form of services whereas Micro Service is a part of SOA. It is an implementation of SOA
* Wiremock, Docker, and Hystrix are some popular Microservices Tools